

IBM RT Advanced Interactive Executive Operating System Version 2.2

---

# AIX Operating System Commands Reference

Volume 2

**Programming Family**

**IBM**

SC23-2081-1

# AIX Operating System Commands Reference

Volume 2

Programming Family



---

## Second Edition (September 1988)

Portions of the code and documentation described in this book were developed at the Electrical Engineering and Computer Sciences Department at the Berkeley Campus of the University of California under the auspices of the Regents of the University of California.

The Rand MH Message Handling System was developed by the Rand Corporation and the University of California.

The Network File System was developed by Sun Microsystems, Inc.

This edition applies to Version 2.2 of the IBM AIX Operating System. Changes are made periodically to the information herein; these changes will be reported in technical newsletters or in new editions of this publication.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM licensed program in this publication is not intended to state or imply that only IBM's licensed program may be used. Any functionally equivalent program may be used instead.

**International Business Machines Corporation provides this manual "as is," without warranty of any kind, either express or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this manual at any time.**

Products are not stocked at the address given below. Requests for copies of this product and for technical information about the system should be made to your IBM authorized RT dealer, your IBM marketing representative, or your IBM authorized remarketer.

A reader's comment form is provided at the back of this publication. If the form has been removed, address comments to IBM Corporation, Department 997, 11400 Burnet Road, Austin, Texas 78758-3493. IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

©Copyright International Business Machines Corporation 1985, 1988

©Copyright INTERACTIVE Systems Corporation 1984

©Copyright AT&T Technologies 1984

---

## Trademarks

The following trademarks apply to this book:

- AIX is a trademark of International Business Machines Corporation.
- DEC and VT100 are trademarks of Digital Equipment Corporation.
- Ethernet is a trademark of XEROX CORPORATION.
- IBM is a registered trademark of International Business Machines Corporation.
- PC-NFS and NFS are trademarks of Sun Microsystems, Inc.
- Proprinter is a trademark of International Business Machines Corporation.
- Quietwriter is a registered trademark of International Business Machines Corporation.
- RT is a registered trademark of International Business Machines Corporation.
- Sun Microsystems is a registered trademark of Sun Microsystems, Inc.
- Tektronics is a trademark of Tektronix, Inc.
- UNIX was developed and licensed by AT&T. It is a registered trademark of AT&T in the United States of America and other countries.
- XEROX is a trademark of XEROX CORPORATION.



---

# About This Book

This book contains reference information on Advanced Interactive Executive (AIX) Operating System commands. It describes the commands you can use and summarizes who can run them, how to run them, what they do, how they read input, how they write output, and how to modify their actions.

## Who Should Use This Book

To use this book, you should be familiar with AIX or UNIX System V commands. If you are not already familiar with AIX or UNIX System V, see *Using the AIX Operating System*. If you are familiar with the commands but need to review how to use the shell and write shell procedures, see “sh” on page 913.

## How To Use This Book

Most of the AIX commands described in this book are in alphabetical order by command name. Some related commands are combined in one description listed with a main or key command. The related commands have an entry with the main command in the table of contents and are listed individually in alphabetical order in the index. If you are having difficulty locating a particular command, check the “Contents” or “Index” sections of this publication.

## Command Information

The “Commands” section begins on page 11. A discussion of a command may include the following information:

<b>Purpose</b>	A single-sentence description of the major function of each command
<b>Syntax</b>	A <i>syntax diagram</i> that shows command line options (For a discussion of how to use this syntax diagram, see “Syntax Diagrams” on page 5.)
<b>Description</b>	A discussion of the command that provides more details about its function and use

---

<b>Flags</b>	A list of command line flags and associated parameters with an explanation of how the flags modify the action of the command
<b>Subcommands</b>	A list of subcommands (for interactive commands) that explains their use
<b>Examples</b>	Specific examples of how you can use the command
<b>Files</b>	A list of files used by the command
<b>Related Information</b>	A list of related commands in this book and related discussions in other books.

For details on other conventions used in this book, see “How to Use the Commands” on page 3.

## A Task Index

“Task Index” on page TASK-1 can help you locate the commands you need to perform specific tasks. It contains lists of commands grouped by task. Next to each command is a description of what it does. To find a command that performs a specific task, locate the task in the table of contexts at the beginning of the task index, go to the indicated page and review the list of commands associated with that task, then select the desired command. For more information about the command, refer to the discussion of the command in the “Commands” section.

## Other Reference Aids

A cross-reference listing of commands and program packages appears in Appendix B, “Program Cross-Reference” on page 1269. Appendix C, “Syntax Diagram Guide” on page 1277 contains a detailed description of how to read syntax diagrams. The standard system devices are described in Appendix A, “AIX Device Table” on page 1267. A “Glossary” of terms appears after the Appendixes, followed by an “Index.”

In addition, a Reader’s Comment Form and Book Evaluation Form are provided at the back of the second volume of this publication. Use the Reader’s Comment Form at any time to give IBM information that may improve the book. After you have become familiar with the book, use the Book Evaluation Form to give IBM specific feedback about the book.

## Japanese Language Support

Appendix D, “Japanese Language Support” on page 1287 contains a list of commands that have not been modified to support Japanese characters.

---

## Special Key Sequences

You can use the AIX Operating System from any of several different display stations, each of which has a different keyboard. In some cases, you must press different keys to perform the same function from different keyboards. Throughout this publication both the function name (for example, INTERRUPT) and the necessary key sequence on the IBM RT system are identified. If you are not using an IBM RT Keyboard, look at your keyboard reference chart to find out which keys on your keyboard produce the same function.

## Prerequisite Information

- *IBM RT Managing the AIX Operating System* provides instructions for performing such system management tasks as adding and deleting user IDs, creating and mounting file systems, repairing file system damage, and managing data communications facilities.
- *IBM RT Using the AIX Operating System* describes using the AIX Operating System commands, working with file systems, developing shell procedures, and using data communications facilities.

## Related Information

- *IBM RT AIX Operating System Programming Tools and Interfaces* describes the programming environment of the AIX Operating System and includes information about using the operating system tools to develop, compile, and debug programs. In addition, this book describes the operating system services and how to take advantage of them in a program. This book also includes a diskette that includes programming examples, written in C language, to illustrate using system calls and subroutines in short, working programs. (Available optionally)
- *IBM RT AIX Operating System Technical Reference* is a four-volume set.
  - System Calls and Subroutines*, describes the system calls and subroutines that a C programmer uses to write programs for the AIX Operating System.
  - Files and Extensions*, contains information about the extensions to the kernel and base operating system, including file formats, special files, and GSL subroutines.
  - VRM Programming Support*, describes the VRM programming environment, including the internal VRM routines, VRM floating-point support, use of the VRM debugger, and the supervisor call instructions that form the Virtual Machine Interface.
  - VRM Device Support*, describes device IPL and configuration, minidisk management, the virtual terminal and block I/O subsystems, as well as the interfaces to VRM device driver and data link control components. This volume also describes the programming

---

conventions for developing your own VRM code and installing it on the system. (Available optionally)

- *IBM RT Using DOS Services* provides step-by-step information for using AIX Operating System **shell**. (Available optionally; packaged with *IBM RT DOS Services Reference*)
- *IBM RT DOS Services Reference* provides reference information about the AIX Operating System **shell**. This book also includes information on sharing DOS files with Personal Computer AT Coprocessor Services, and on the differences between PC DOS and **shell**. (Available optionally; packaged with *IBM RT Using DOS Services*)
- *IBM RT C Language Guide and Reference* provides guide information for writing, compiling, and running C language programs and includes reference information about C language data structures, operators, expressions, and statements. (Available optionally)
- *IBM RT Messages Reference* lists messages displayed by the IBM RT and explains how to respond to the messages.
- *IBM RT AIX Operating System Text Formatting Guide* describes the functions and capabilities of NROFF and TROFF to perform text processing tasks. (Available optionally)
- *IBM RT Bibliography and Master Index* provides brief descriptive overviews of the books and tutorial program that support the IBM RT hardware and the AIX Operating System. In addition, this book contains an index to the RT and AIX Operating System library.

See *IBM RT Bibliography and Master Index* for order numbers of IBM RT publications and diskettes.

## Ordering Additional Copies of This Book

To order additional copies of this publication (without program diskettes), use either of the following sources:

- To order from your IBM representative, use Order Number SBOF-1814.
- To order from your IBM dealer, use Part Number 27F4354.

A binder is included with the order. For information on ordering the binder and manual separately, contact your IBM representative or your IBM dealer.

---

# Contents

<b>VOLUME 1</b> .....	<b>1</b>
<b>How to Use the Commands</b> .....	<b>3</b>
Command Input and Output .....	4
File Name Substitution .....	4
Syntax Diagrams .....	5
Command, Flag, and Parameter Notation .....	8
<b>Commands</b> .....	<b>11</b>
<b>acct/*</b> .....	13
<b>chargefee</b> .....	14
<b>ckpacct</b> .....	14
<b>dodisk</b> .....	14
<b>lastlogin</b> .....	15
<b>monacct</b> .....	15
<b>nulladm</b> .....	15
<b>prctmp</b> .....	15
<b>prdaily</b> .....	15
<b>prtacct</b> .....	16
<b>remove</b> .....	16
<b>shutacct</b> .....	16
<b>startup</b> .....	16
<b>turnacct</b> .....	16
<b>acctems</b> .....	18
<b>acctcom</b> .....	20
<b>acctcon</b> .....	24
<b>acctcon1</b> .....	24
<b>acctcon2</b> .....	25
<b>acctdisk, acctdusg</b> .....	26
<b>acctmerg</b> .....	28
<b>acctpre</b> .....	30
<b>acctpre1</b> .....	30
<b>acctpre2</b> .....	31
<b>accton</b> .....	31
<b>actman</b> .....	32
<b>adb</b> .....	33
<b>admin</b> .....	41
<b>ali</b> .....	48
<b>anno</b> .....	50

---

<b>ap</b>	53
<b>ar</b>	55
<b>arithmetic</b>	59
<b>as</b>	61
<b>at, batch</b>	63
<b>audit</b>	67
<b>auditapp</b>	69
<b>auditbin</b>	71
<b>auditpr</b>	73
<b>auditselect</b>	76
<b>auditstream</b>	78
<b>auditwrite</b>	80
<b>awk</b>	81
<b>back</b>	87
<b>backup</b>	88
<b>banner</b>	94
<b>basename, dirname</b>	95
<b>bc</b>	97
<b>bdiff</b>	102
<b>bellmail</b>	104
<b>bffcreate</b>	108
<b>bfs</b>	110
<b>biod</b>	114
<b>biodd_cfg</b>	115
<b>bj</b>	117
<b>bs</b>	118
<b>burst</b>	129
<b>cal</b>	132
<b>calendar</b>	134
<b>cat</b>	137
<b>cb</b>	139
<b>cc</b>	140
<b>cd</b>	150
<b>cdc</b>	152
<b>cflow</b>	154
<b>chgrp</b>	156
<b>chkcomp</b>	158
<b>chmod</b>	160
<b>chngstate</b>	164
<b>installc</b>	166
<b>updatec</b>	167
<b>chown</b>	169
<b>chparm</b>	171
<b>chroot</b>	172
<b>chtcb</b>	174
<b>clri</b>	175

---

<b>cmp</b>	177
<b>col</b>	179
<b>comb</b>	181
<b>comm</b>	183
<b>comp</b>	185
<b>confer</b>	189
<b>config</b>	194
<b>conflict</b>	196
<b>connect</b>	198
<b>cp</b>	202
<b>cpio</b>	205
<b>cpp</b>	210
<b>craps</b>	214
<b>crash</b>	215
<b>cron</b>	220
<b>crontab</b>	222
<b>csch</b>	225
<b>csplit</b>	252
<b>ct</b>	254
<b>ctab</b>	257
<b>ctags</b>	261
<b>cu</b>	263
<b>cut</b>	269
<b>cvid</b>	272
<b>Cvt</b>	274
<b>cw, checkcw</b>	275
<b>cxref</b>	279
<b>date</b>	281
<b>dbx</b>	284
<b>dc</b>	295
<b>dcopy</b>	299
<b>dd</b>	301
<b>defkey</b>	306
<b>del</b>	308
<b>delta</b>	310
<b>deroff</b>	313
<b>devices</b>	315
<b>devnm</b>	316
<b>df</b>	318
<b>diff</b>	320
<b>diff3</b>	323
<b>diffmk</b>	326
<b>dircmp</b>	328
<b>diskusg</b>	330
<b>display</b>	332
<b>dist</b>	336

---

<b>domainname</b>	340
<b>dos</b>	341
<b>dosdel</b>	345
<b>dosdir</b>	346
<b>dosread</b>	348
<b>doswrite</b>	350
<b>dp</b>	352
<b>dsipc</b>	354
<b>dsldxprof</b>	355
<b>dspcat</b>	357
<b>dspmsg</b>	359
<b>dsstate</b>	361
<b>dsxlate</b>	363
<b>du</b>	364
<b>dump</b>	366
<b>dumpfmt</b>	368
<b>echo</b>	369
<b>ed</b>	371
<b>edconfig</b>	385
<b>edit</b>	387
<b>env</b>	393
<b>eqn, neqn, checkeq</b>	395
<b>errdead</b>	397
<b>errdemon</b>	398
<b>errpt, errpd</b>	400
<b>errstop</b>	404
<b>errupdate</b>	405
<b>ex</b>	407
<b>expr</b>	412
<b>factor</b>	416
<b>ff</b>	417
<b>file</b>	420
<b>find</b>	422
<b>fish</b>	427
<b>fmt</b>	428
<b>folder</b>	429
<b>folders</b>	433
<b>format</b>	436
<b>fortune</b>	437
<b>forw</b>	438
<b>fptype</b>	444
<b>fsck, dfsck</b>	445
<b>fsdb</b>	450
<b>fuser</b>	455
<b>fwtmp</b>	457
<b>acctwtmp</b>	458

---

<b>wtmpfix</b> .....	458
<b>gdev</b> .....	460
<b>hpd</b> .....	460
<b>erase</b> .....	461
<b>hardcopy</b> .....	461
<b>tekset</b> .....	461
<b>td</b> .....	461
<b>ged</b> .....	463
<b>gencat</b> .....	470
<b>gend</b> .....	475
<b>get</b> .....	477
<b>getopt</b> .....	485
<b>gettext</b> .....	488
<b>getty</b> .....	490
<b>graph</b> .....	494
<b>graphics</b> .....	497
<b>greek</b> .....	499
<b>grep</b> .....	501
<b>groups</b> .....	506
<b>gutil</b> .....	508
<b>bel</b> .....	509
<b>cvrtopt</b> .....	509
<b>gd</b> .....	510
<b>gtop</b> .....	510
<b>pd</b> .....	510
<b>ptog</b> .....	510
<b>quit</b> .....	510
<b>remcom</b> .....	510
<b>whatis</b> .....	511
<b>yoo</b> .....	511
<b>hangman</b> .....	512
<b>help</b> .....	513
<b>hp</b> .....	514
<b>hyphen</b> .....	516
<b>id</b> .....	517
<b>inc</b> .....	518
<b>init</b> .....	521
<b>install</b> .....	524
<b>install-mh</b> .....	527
<b>installp</b> .....	529
<b>inusave</b> .....	531
<b>inurecv</b> .....	532
<b>inurest</b> .....	533
<b>ckprereq</b> .....	533
<b>mvmd</b> .....	534
<b>iperm</b> .....	537

<b>ipcs</b>	539
<b>ipctable</b>	544
<b>istat</b>	545
<b>join</b>	547
<b>keyboard</b>	551
<b>kill</b>	552
<b>killall</b>	555
<b>ld</b>	557
<b>lex</b>	562
<b>li</b>	567
<b>line</b>	574
<b>link, unlink</b>	575
<b>lint</b>	577
<b>ln</b>	581
<b>locator</b>	583
<b>login</b>	584
<b>loginx</b>	587
<b>logname</b>	589
<b>logout</b>	590
<b>lorder</b>	591
<b>lp</b>	593
<b>ls</b>	595
<b>VOLUME 2</b>	<b>601</b>
<b>m4</b>	603
<b>mail, Mail</b>	608
<b>mailstats</b>	623
<b>make</b>	625
<b>makedbm</b>	632
<b>makekey</b>	634
<b>man</b>	635
<b>mark</b>	637
<b>mdrc</b>	640
<b>mesg</b>	642
<b>mhl</b>	643
<b>mhmail</b>	646
<b>mhpath</b>	648
<b>minidisks</b>	650
<b>mkcatdefs</b>	651
<b>mkdir</b>	657
<b>mkfs</b>	658
<b>mknod</b>	661
<b>mm, checkmm</b>	663
<b>mmt, checkmm</b>	666
<b>moo</b>	668

---

mount	669
mountd	674
msgchk	675
msh	677
mv	679
mmdir	682
ncheck	683
ndtable	685
newform	686
newgrp	689
news	691
next	694
nfsd	696
nfsstat	697
nice	699
nl	701
nm	705
nohup	707
nroff, troff	709
number	721
od	723
on	726
open	728
pack	730
pcat	731
unpack	731
packf	733
passwd	735
paste	736
pcnfs	739
pdisable, phold	741
pg	744
pick	748
piobe	753
portmap	757
post	758
pr	761
prev	765
print	767
prof	773
profiler	775
prfld	776
prfstat	776
prfdc, prfsnap	776
prfpr	776
prompter	778

---

<b>proto</b>	780
<b>prs</b>	781
<b>ps</b>	786
<b>pstart, penable, pshare, pdelay</b>	791
<b>ptx</b>	794
<b>puttext</b>	796
<b>pwck</b>	798
<b>pwd</b>	800
<b>pwtable</b>	801
<b>qdaemon</b>	802
<b>quiz</b>	803
<b>rc</b>	806
<b>rcvdist</b>	808
<b>rcvpack</b>	810
<b>rcvstore</b>	812
<b>rcvtty</b>	815
<b>refile</b>	817
<b>regcmp</b>	820
<b>repl</b>	821
<b>restore</b>	826
<b>rex</b>	832
<b>rm</b>	833
<b>rmail</b>	836
<b>rmdel</b>	837
<b>rmdir</b>	838
<b>rmf</b>	839
<b>rmm</b>	841
<b>rpcgen</b>	843
<b>rpcinfo</b>	845
<b>rstatd</b>	847
<b>runacct</b>	848
<b>runcat</b>	852
<b>rup</b>	854
<b>rusers</b>	856
<b>rusersd</b>	858
<b>rwall</b>	859
<b>rwalld</b>	861
<b>sact</b>	862
<b>sadc</b>	863
<b>sa1</b>	864
<b>sa2</b>	864
<b>sag</b>	865
<b>sar</b>	867
<b>scan</b>	871
<b>sccsdiff</b>	874
<b>sdb</b>	875

---

<b>sdiff</b>	883
<b>secure</b>	885
<b>sed</b>	887
<b>send</b>	893
<b>sendmail</b>	897
<b>setdma</b>	910
<b>setmnt</b>	911
<b>sh</b>	913
<b>shell</b>	938
<b>shlib</b>	939
<b>show</b>	942
<b>showmount</b>	945
<b>shutdown</b>	946
<b>size</b>	949
<b>skulker</b>	951
<b>sleep</b>	952
<b>slocal</b>	954
<b>sno</b>	956
<b>sort</b>	958
<b>sortm</b>	965
<b>sound</b>	967
<b>spell</b>	969
<b>spline</b>	972
<b>split</b>	974
<b>splp</b>	975
<b>spost</b>	978
<b>spray</b>	981
<b>sprayd</b>	983
<b>stat</b>	984
<b>strip</b>	1017
<b>stty</b>	1018
<b>su</b>	1026
<b>sum</b>	1029
<b>sync</b>	1030
<b>sysck</b>	1031
<b>syslogd</b>	1037
<b>tab, untab</b>	1040
<b>tabs</b>	1041
<b>tail</b>	1044
<b>tapechk</b>	1047
<b>tar</b>	1048
<b>tbl</b>	1053
<b>tc</b>	1056
<b>tctl</b>	1058
<b>tee</b>	1060
<b>termdef</b>	1062

---

<b>test</b>	1064
<b>tic</b>	1067
<b>time</b>	1068
<b>timex</b>	1069
<b>tlog</b>	1071
<b>tlogger</b>	1072
<b>toc</b>	1074
<b>dtoc</b>	1074
<b>ttoc</b>	1075
<b>vtoc</b>	1075
<b>touch</b>	1077
<b>tplot</b>	1079
<b>tput</b>	1081
<b>tr</b>	1083
<b>trace</b>	1086
<b>trcrpt</b>	1091
<b>trcstop</b>	1093
<b>trcupdate</b>	1094
<b>trdiag</b>	1097
<b>true</b>	1099
<b>tsh</b>	1100
<b>tsort</b>	1102
<b>ttt</b>	1104
<b>tty</b>	1105
<b>turnon</b>	1107
<b>tvi</b>	1108
<b>ugtable</b>	1109
<b>umask</b>	1110
<b>umount, unmount</b>	1112
<b>uname</b>	1114
<b>unget</b>	1116
<b>uniq</b>	1118
<b>units</b>	1119
<b>updatep</b>	1122
<b>inudocm</b>	1125
<b>inuupdt</b>	1127
<b>users, adduser</b>	1129
<b>uucpadm</b>	1133
<b>uucheck</b>	1137
<b>uucico</b>	1139
<b>uucleanup</b>	1141
<b>uucp</b>	1144
Path Names Used with uucp	1145
Source and Destination File Names	1145
Permissions	1146
<b>uulog</b>	1149

---

<b>uuname</b>	1151
<b>uupick</b>	1153
File-Handling Options	1154
<b>uusched</b>	1156
<b>uustat</b>	1158
<b>uuto</b>	1162
<b>uutry, Uutry, uukick</b>	1164
<b>uux</b>	1166
<b>uuxqt</b>	1172
<b>val</b>	1175
<b>varyoff</b>	1177
<b>varyon</b>	1180
<b>vc</b>	1182
<b>verify</b>	1186
<b>vi, vedit, view</b>	1187
<b>vmh</b>	1203
<b>vrn2rtfont</b>	1205
<b>vrnconfig</b>	1206
<b>wall</b>	1208
<b>watch</b>	1209
<b>wc</b>	1211
<b>what</b>	1213
<b>whatnow</b>	1215
<b>who</b>	1219
<b>whom</b>	1222
<b>write</b>	1225
<b>writesrv</b>	1230
<b>wump</b>	1231
<b>xargs</b>	1232
<b>xdbx</b>	1236
<b>yacc</b>	1237
<b>ypbind</b>	1239
<b>ypcat</b>	1241
<b>ypinit</b>	1243
<b>ypmatch</b>	1245
<b>yppasswd</b>	1247
<b>yppasswdd</b>	1249
<b>yppoll</b>	1251
<b>yppush</b>	1252
<b>ypset</b>	1254
<b>ypserv</b>	1256
<b>ypwhich</b>	1258
<b>ypxfr</b>	1260
<b>300</b>	1262
<b>4014</b>	1264
<b>450</b>	1265

---

<b>Appendix A. AIX Device Table</b>	<b>1267</b>
<b>Appendix B. Program Cross-Reference</b>	<b>1269</b>
<b>Appendix C. Syntax Diagram Guide</b>	<b>1277</b>
<b>Appendix D. Japanese Language Support</b>	<b>1287</b>
<b>Glossary</b>	<b>1291</b>
<b>Task Index</b>	<b>TASK-1</b>
<b>Index</b>	<b>INDEX-1</b>

---

## Figures

1.	SCCS Header Flags	44
2.	SID Determination	481
3.	Mailbox Commands	611
4.	Mail Editor Commands	617
5.	Binary Options	619
6.	Valued Options	620
7.	Delta Table Keywords	782
8.	Header Flag Keywords	783
9.	Other Keywords	784
10.	Configuration Options	905
11.	<b>tbl</b> Column and Item Specifiers	1054
12.	Configuration File Parameters	1129
13.	AIX Standard Devices (Special Files)	1268



---

**VOLUME 2**



---

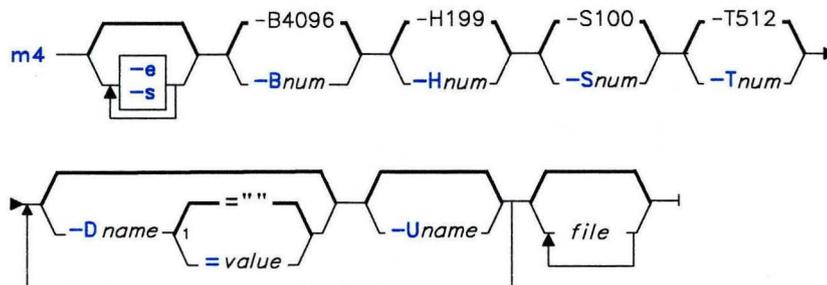
## m4

---

### Purpose

Preprocesses files, expanding macro definitions.

### Syntax



OL805434

### Description

The `m4` command is a macroprocessor used as a preprocessor for C and other languages. You can use it to process built-in macros or user-defined macros. Each *file* is processed in order. If you do not specify a *file* or if you give a minus (-) as a file name, `m4` reads standard input. It writes the processed macros to standard output. Macro calls follow the form:

```
macroname(argument . . . )
```

The left parenthesis must immediately follow `macroname`. If the left parenthesis does not follow the name of a defined macro, `m4` reads it as a macro call with no arguments. Macro names consist of ASCII alphabetic letters, digits, and the underscore character (`_`). Extended characters are not allowed in macro names. The first character cannot be a digit.

While collecting arguments, `m4` ignores unquoted leading blanks, tabs, and new-line characters. Use single quotation marks to quote strings. The value of a quoted string is the string with the quotation marks stripped off.

When **m4** recognizes a macro, it collects arguments by searching for a matching right parenthesis. If you supply fewer arguments than appear in the macro definition, **m4** considers the trailing arguments in the definition to be null. Macro evaluation proceeds normally during the collection of the arguments. All commas or right parentheses within the value of a nested call are translated literally; they do not need an escape character or quotation marks. After collecting arguments, **m4** pushes the value of the macro back onto the input stream and scans again.

## Built-in Macros

The **m4** command makes available the following built-in macros. You may redefine them, but you will lose the original meaning. The values of these macros are null unless otherwise stated:

- define**(*name,new\_name*) Replaces the macro *name* with the value of *new\_name*. The *new\_name* string can take the form  $\$n \dots$  (where *n* is a digit). In this case, each occurrence of *n* in the replacement text is replaced by the *n*-th argument of *name*.  $\$0$  is the name of the macro. The null string replaced missing arguments. The number of arguments replaces  $\#\$ . A comma-separated list of all arguments replaces  $\*\$ .  $\@\$  acts like  $\*\$ , but each argument is quoted with the current quotation character (see **changequote**).
- undefine**(*name*) Removes the definition of *name*.
- defn**(*name . . .*) Returns the quoted definition of *name*.
- pushdef**(*name,new\_name*) Redefines *name* with *new\_name* as in *define*, but save any previous definition.
- popdef**(*name . . .*) Removes the current definition of *name* and returns to the previous definition, if one existed.
- ifdef**(*name,true,[false]*) Returns the value of *true* only if *name* is defined, otherwise return *false*. If you do not supply *false*, its value is null.
- Note:** The word **unix** is predefined.
- shift**(*argument . . .*) Returns all but the first argument. The other arguments are quoted and pushed back with commas in between. The quoting nullifies the effect of the extra scan that will subsequently be performed.
- changequote**(*L,R*) Changes quote symbols to *L* and *R*. The symbols can be up to 5 bytes long. **changequote** without arguments restores the original values ( ` ' ).

---

<b>changecom</b> ( <i>Lcom</i> , <i>Rcom</i> )	Changes left and right comment markers from the default # and new-line character to <i>Lcom</i> and <i>Rcom</i> . With no arguments, the comment mechanism is disabled. With one argument, the left marker becomes the parameter and the right marker becomes a new-line character. With two arguments, both markers are affected. Comment markers can be up to 5 bytes long.
<b>divert</b> ( <i>num</i> )	Changes the current output stream to stream <i>num</i> . There are 10 output streams, numbered 0-9. The final output is the concatenation of the streams in numerical order. Initially, stream 0 is the current stream. <b>m4</b> discards output diverted to a stream other than 0-9.
<b>undivert</b> ( <i>num</i> . . . )	Causes immediate output of text from the specified diversions (or all diversions if there is no argument). Text may be undiverted into another diversion. Undiverting discards the diverted text.
<b>divnum</b>	Returns the value of the current output stream.
<b>dnl</b>	Reads and discards characters up to and including the next new-line character.
<b>ifelse</b> ([ <i>string1</i> , <i>string2</i> , <i>true</i> ,[ <i>false</i> ] . . . )	If <i>string1</i> and <i>string2</i> are the same, then the value is <i>true</i> . If they are not and if there are more than four arguments, <b>m4</b> repeats the process with the additional arguments (4, 5, 6, and 7). Otherwise, the value is either <i>false</i> or null if you provide no value for <i>false</i> .
<b>incr</b> ( <i>num</i> )	Returns the value of its argument incremented by 1.
<b>decr</b> ( <i>num</i> )	Returns the value of its argument decreased by 1.
<b>eval</b> ( <i>expr</i> [, <i>num1</i> [, <i>num2</i> ]])	Evaluates its first argument as an arithmetic expression, using 32-bit arithmetic. The operators you can use include +, -, *, /, %, ^ (exponentiation), bitwise &, !, ~, and ^ relationals, and parentheses. Octal and hex numbers can be specified as in C. <i>num1</i> specifies the radix for the result of the expression. The default radix is 10. The optional <i>num2</i> specifies the minimum number of digits in the result.
<b>len</b> ( <i>string</i> )	Returns the number of bytes in <i>string</i> .
<b>dlen</b> ( <i>string</i> )	Returns the number of displayable characters in <i>string</i> ; that is, 2-byte extended characters are counted as one displayable character.

<b>index</b> ( <i>s1,s2</i> )	Returns the position in the string <i>s1</i> where the string <i>s2</i> begins (zero origin), or -1 if the second parameter does not occur.
<b>substr</b> ( <i>string,position,num</i> )	Returns a substring of <i>string</i> . The beginning of the substring is selected with <i>position</i> , and <i>num</i> indicates the length of the substring. Without <i>num</i> , the substring includes everything to the end of the first string.
<b>translit</b> ( <i>string,from,to</i> )	Transliterates the characters in <i>string</i> from the set given by <i>from</i> to the set given by <i>to</i> . No abbreviations are permitted. Two-byte extended characters are correctly mapped into the corresponding replacement characters.
<b>include</b> ( <i>file</i> )	Returns the contents of <i>file</i> or displays an error message if it cannot access the file.
<b>sinclude</b> ( <i>file</i> )	Returns the contents of <i>file</i> , but it gives no error message if <i>file</i> is inaccessible.
<b>syscmd</b> ( <i>command</i> )	Runs the AIX <i>command</i> . No value is returned.
<b>sysval</b>	Returns the return code from the last call to <i>syscmd</i> .
<b>maketemp</b> ( . . . <b>XXXXX</b> . . . )	Replaces <b>XXXXX</b> in its argument with the current process ID number.
<b>m4exit</b> ( <i>value</i> )	Exits from <b>m4</b> immediately, returning the specified exit <i>value</i> (the default is 0).
<b>m4wrap</b> ( <i>lastmacro</i> )	Runs <i>lastmacro</i> after reading the end-of-file character. For example: <code>m4wrap(`cleanup()')</code> runs the <code>cleanup</code> macro at the end of <b>m4</b> .
<b>errprint</b> ( <i>message</i> )	Includes <i>message</i> on the diagnostic output file.
<b>dumpdef</b> ([ <i>name</i> . . . ])	Writes to standard output the current names and definitions for the named items or for all if no arguments are provided.
<b>traceon</b> ( <i>macro</i> )	Turns on tracing for <i>macro</i> . If none is named, tracing is turned on for all macros.
<b>traceoff</b> ( <i>macro</i> . . . )	Turns off trace globally and for any <i>macro</i> specified. Macros specifically traced by <b>traceon</b> can be untraced only by specific calls to <b>traceoff</b> .

---

## Flags

- Bnum**            Makes *num* the size of the push-back and parameter collection buffers (the default is 4096).
  - e**                Operates interactively. Interrupts are ignored and the output is not buffered.
  - Hnum**            Makes *num* the size of the symbol table hash array (the default is 199). The size must be a prime number.
  - s**                Enables the line sync output for the C preprocessor (`#line . . .`).
  - Snum**            Makes *num* the size of the call stack (the default is 800 slots). Macros take three slots, and nonmacro arguments take one.
  - Tnum**            Makes *num* the size of the token buffer (the default is 512 bytes).
- The preceding flags must appear before any file names and before any **-D** or **-U** flags.
- Dname[=val]**    Define *name* as *val*. If *val* is not specified, *name* becomes null.
  - Uname**            Undefines a *name* previously defined with the **-D** flag.

## Example

To preprocess a C language program with **m4** and compile it:

```
m4 prog.m4 >prog.c
cc prog.c
```

## Related Information

The following commands: “**cc**” on page 140 and “**cpp**” on page 210.

“Overview of International Character Support” in *Managing the AIX Operating System*.

# mail, Mail

---

## mail, Mail

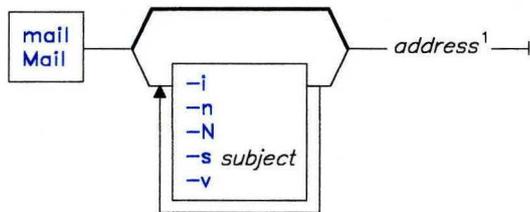
---

### Purpose

Sends and receives mail.

### Syntax

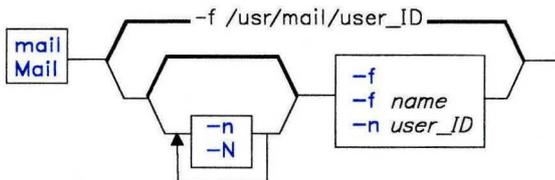
#### Sending Mail:



<sup>1</sup>See `sendmail` for address formats.

AJ2FL262

#### Handling Mail:



AJ2FL260

### Description

The `mail` program allows you to:

- Compose a message and send it
- Receive a message and look at it
- Store received messages in your mailbox, or in folders
- Discard messages.

To send a message to one or more persons, enter **mail** on the command line with arguments that are the network addresses of the people to receive the message. When **mail** starts, you can type the message using an editor similar to **ed**. When you are finished with the message, press the **Enter** key at the end of a line, and use a **Ctrl-D** (EOF) sequence at the beginning of the next line to exit the editor and send the message.

When you have messages in your mailbox, the system displays a message to tell you. The default message is:

```
[YOU HAVE NEW MAIL]
```

To look at the contents of your mailbox, enter the **mail** command without arguments on the command line. The program displays a listing of the messages in your mailbox and allows you to look at them, reply to them or dispose of them.

## Reading Incoming Mail

To receive and read incoming mail, use **mail** with no arguments:

```
mail
```

The **mail** command then checks your system mailbox (**/usr/mail/user-id**) and displays a one-line entry for each message in the system mailbox similar to:

```
"/usr/mail/geo": 2 messages 2 new
>N 1 amy      Thu Sep 17 14:36 13/359 "Dept Meeting"
  N 2 amy      Thu Sep 17 16:28 13/416 "Dept Meeting Delayed"
&
```

The **>** symbol indicates the **current message**, or the message that commands act on if you do not specify a message number or list of message numbers. The other fields, in order, in the listing represent:

1. Message number
2. User address of the sender
3. Date the message was sent
4. Size of the message in lines/characters
5. The subject of the message (if one was included in the message).

From the **mail** command prompt **&**, you can enter commands to look at, reply to, save, discard, or otherwise manage the contents of the mailbox. To display a summary of some of the commands that you can use to handle mail in your mailbox, enter **?** at the **mail** command prompt. For more information on those commands and information on additional commands, refer to Figure 3 on page 611.

Many mailbox commands allow you to specify groups of messages upon which to perform the command. Commands that allow groups of messages use the parameter *msg\_lst* in the command format. For example, the format of the **f** command (display information about messages) appears as:

```
& f msg_lst
```

## mail, Mail

---

In this format *msg-*lst** can be one of the following:

- One or more message numbers separated by spaces  
`& f 1 2 4 7`
- A range of message numbers indicated by the first and last numbers in the range separated by a dash  
`& f 2-5`  
is the same as,  
`& f 2 3 4 5`
- One or more addresses separated by spaces to apply the command to messages received from those addresses,  
`& f amy geo@zeus`

The characters entered for an address need not match the address exactly. They must only be contained in the address field of the messages in either upper- or lower-case. Therefore, the request for address `amy` matches all of the following addresses (and many others):

- amy
- AmY
- amy@zeus
- hamy

- A string, preceded by a slash, to match against the **Subject:** field of the messages,  
`& f /meet`

applies the command to all messages whose **Subject:** field contains the letters `meet` in upper- or lower-case. The characters entered for a match pattern do not need to match the **Subject:** field exactly. They must only be contained in the **Subject:** field of the messages in either upper- or lower-case. Therefore, the request for subject `meet` matches all of the following subjects (and many others):

- Meeting on Thursday
- Come to meeting tomorrow
- MEET ME IN ST. LOUIS

The special character `*` (asterisk) addresses all messages, `^` (caret) addresses the first message, and `$` (dollar sign) addresses the last message.

The following table lists the **mail** commands and describes their functions.

Command	Function
=	Echoes the number of the current message.
#	Comment character for writing comments in mail script files.
- <i>n</i>	Goes to the previous message and displays it. If given a number argument of <i>n</i> , goes to the <i>n</i> th previous message and displays it.
?	Displays a brief summary of commands.
! <i>sh_cmd</i>	Executes the AIX shell command specified by <i>sh_cmd</i> .
alias	(a) With no arguments, displays all currently defined aliases. With one argument, displays alias. With more than one argument, creates a new or changes an old alias.
alternates <i>alt_list</i>	(alt) The <b>alternates</b> command is useful if you have accounts on several machines. Use it to inform <b>mail</b> that the addresses listed in <i>alt_list</i> all refer to you. Then, when you <b>reply</b> to messages, <b>mail</b> does not send a copy of the message to any of the addresses given in <i>alt_list</i> . If you enter the <b>alternates</b> command with no argument, <b>mail</b> displays the current set of alternate names.
chdir <i>dir</i>	(cd) Changes your working directory to the directory <i>dir</i> . If no directory is given, it changes to your login directory.
copy <i>msg_lst file</i>	(c, co) Appends each message in <i>msg_lst</i> in turn to the end of <i>file</i> . Displays the file name in quotes, followed by the line count and character count, on the user's terminal. Does not delete any messages when you quit.
delete <i>msg_lst</i>	(d) Marks the messages in <i>msg_lst</i> to be deleted when you quit <b>mail</b> . Deleted messages are not saved in <b>mbox</b> , nor are they available for most other commands. However, you can restore messages that you have deleted while in the same mailbox session (see the <b>undelete</b> mailbox command).

Figure 3 (Part 1 of 6). Mailbox Commands

## mail, Mail

---

Command	Function
<b>discard</b> [ <i>fld_lst</i> ]	<b>(di)</b> Identical to the <b>ignore</b> command.
<b>dp</b>	Deletes the current message and displays the next message. If there is no next message, <b>mail</b> displays the message at EOF.
<b>dt</b>	Identical to the <b>dp</b> command.
<b>echo</b> <i>string</i>	Displays the character string <i>string</i> on the command line.
<b>edit</b> <i>msg</i>	<b>(e)</b> Activates the editor that you define with the <b>set EDITOR=</b> statement and loads message <i>msg</i> into the editor. When you exit the editor, the saved message is replaced in the mailbox being processed.
<b>exit</b>	<b>(ex or x)</b> Exits to the shell without changing the mailbox being processed. The mailbox returns to the condition that it was when <b>mail</b> was started. Messages marked to be deleted are not deleted.
<b>file</b> [ <i>name</i> ]	<b>(fi)</b> Identical to the <b>folder</b> command.
<b>folder</b> [ <i>name</i> ]	<b>(fo)</b> Switches to a new mail file or folder. With no arguments, displays the name of the mailbox that you are currently reading. If an argument is included, it stores the current mailbox with changes (such as messages deleted) and reads in the new mailbox specified by the <i>name</i> parameter. Some special conventions are recognized for the <i>name</i> : <ul style="list-style-type: none"><li>• # refers to the previous file</li><li>• % refers to the system mailbox</li><li>• &amp; refers to your personal mailbox (<b>\$HOME/mbox</b>)</li><li>• <i>name</i> refers to a file in your folder directory.</li></ul>
<b>folders</b>	Lists the names of the folders in your folder directory.
<b>from</b> <i>msg_lst</i>	<b>(f)</b> Displays the headings of messages in <i>msg_lst</i> .
<b>group</b>	<b>(g)</b> Identical to the <b>alias</b> command.

Figure 3 (Part 2 of 6). Mailbox Commands

Command	Function
<b>headers</b>	<b>(h)</b> Lists the headings in the current group of messages (each group of messages contains 20 messages by default; change this with the <b>set screen =</b> statement).
<b>help</b>	Identical to question mark (?).
<b>hold</b> <i>msg_lst</i>	<b>(ho)</b> Marks each message in <i>msg_lst</i> to be saved in your system mailbox instead of in <b>mbox</b> . Does not override the <b>delete</b> command.
<b>if</b> <i>condition</i> <b>else</b> <b>endif</b>	Construction for conditional execution of <b>mail</b> commands. Commands following <b>if</b> are executed if <i>condition</i> is true. Commands following <b>else</b> are executed if <i>condition</i> is not true. The <b>else</b> is not required. The <b>endif</b> ends the construction and is required. The <i>condition</i> can be <b>receive</b> (receiving mail) or <b>send</b> (sending mail).
<b>ignore</b> [ <i>fld_lst</i> ]	Adds the header fields in <i>fld_lst</i> to the list of fields to be ignored. Ignored fields are not displayed when you look at a message with the <b>t</b> or <b>p</b> commands. Use this command to suppress machine-generated header fields. Use the <b>Type</b> and <b>Print</b> commands to print a message in its entirety, including ignored fields. If <b>ignore</b> is executed with no arguments, it lists the current set of ignored fields.
<b>list</b>	<b>(l)</b> Displays a list of valid <b>mail</b> commands.
<b>local</b>	Lists other names for the local host.
<b>mail</b> <i>addr_lst</i>	<b>(m)</b> Activates the mail editor to allow you to create and send a message to people specified in <i>addr_lst</i> .
<b>mbox</b> <i>msg_lst</i>	Indicates that the messages in <i>msg_lst</i> be sent to your personal mailbox when you quit. This operation is the default action for messages that you have looked at if you are looking at your system mailbox and the <b>hold</b> option is not set.
<b>more</b> <i>msg_lst</i>	<b>(mo)</b> Displays the messages in <i>msg_lst</i> using the defined pager program to control display to the screen.
<b>More</b> <i>msg_lst</i>	<b>(Mo)</b> Like <b>more</b> but also displays ignored header fields. See <b>more</b> and <b>ignore</b> .

Figure 3 (Part 3 of 6). Mailbox Commands

## mail, Mail

---

Command	Function
<b>new</b> <i>msg_lst</i>	Identical to the <b>unread</b> command.
<b>New</b> <i>msg_lst</i>	Identical to the <b>Unread</b> command.
<b>next</b> [ <i>msg</i> ]	<b>(n)</b> Makes the next message in the mailbox the current message and displays that message. With an argument list, it displays the next matching message.
<b>page</b> <i>msg_lst</i>	<b>(pa)</b> Identical to the <b>more</b> command.
<b>Page</b> <i>msg_lst</i>	<b>(Pa)</b> Identical to the <b>More</b> command.
<b>preserve</b>	<b>(pre)</b> Identical to the <b>hold</b> command.
<b>print</b> <i>msg_lst</i>	<b>(p)</b> Displays the messages in <i>msg_lst</i> .
<b>Print</b> <i>msg_lst</i>	<b>(P)</b> Like <b>print</b> but also displays ignored header fields. See <b>print</b> and <b>ignore</b> .
<b>quit</b>	<b>(q)</b> Ends the session and returns to the shell. Before ending, <b>mail</b> saves all messages that have not been deleted or saved in your personal mailbox ( <b>\$HOME/mbox</b> ). It keeps all messages marked with <b>hold</b> or <b>preserve</b> and those messages that have not been looked at, in the system mailbox. It removes all other messages from the system mailbox. If given while editing a mailbox file with the <b>-f</b> flag, then the edit file is saved with changes. If the edit file cannot be saved, <b>mail</b> does not exit. Use the <b>exit</b> command to exit without saving the changes.
<b>reply</b> <i>msg</i>	Allows you to create and send mail to the people who sent and received the message specified in <i>msg</i> .
<b>Reply</b> <i>msg</i>	Allows you to create and send mail only to the person who sent the message specified in <i>msg</i> .
<b>respond</b> <i>msg</i>	Identical to the <b>reply</b> command.
<b>Respond</b> <i>msg</i>	Identical to the <b>Reply</b> command.

Figure 3 (Part 4 of 6). Mailbox Commands

---

Command	Function
<b>retain</b> [ <i>fld_lst</i> ]	Adds the header fields in <i>fld_lst</i> to the list of fields to be retained. Retained fields are displayed when you look at a message with the <b>t</b> or <b>p</b> commands. Use this command to define which header fields you want displayed. Use the <b>Type</b> and <b>Print</b> commands to print a message in its entirety, including fields that are not retained. If <b>retain</b> is executed with no arguments, it lists the current set of retained fields.
<b>save</b> <i>msg_lst file</i>	( <b>s</b> ) Appends the messages specified in <i>msg_lst</i> to <i>file</i> . Displays the file name and the size of the file when the operation is complete. If you save a message to a file, that message is not returned to the system mailbox nor saved in your personal mailbox when you quit the <b>mail</b> program.
<b>set</b> [ <i>option</i> ]	( <b>se</b> ) With no arguments, prints all variable values. Otherwise, sets an option as specified in <i>option</i> . The <i>option</i> field can be either the name of a <b>binary</b> option (an option that is either set or not set) or a statement of the form:  option=value  That assigns a value to a valued option. Binary and valued options are described later in this command description.
<b>shell</b>	( <b>sh</b> ) Invokes an interactive version of the shell.
<b>size</b> <i>msg_lst</i>	Displays the sizes in lines/characters of the messages in <i>msg_lst</i> .
<b>source</b> <i>file</i>	( <b>so</b> ) Reads <b>mail</b> commands from <i>file</i> .
<b>top</b> <i>msg_lst</i>	Displays the top few lines of the messages specified by <i>msg_lst</i> . The number of lines displayed is determined by the valued option <b>toplines</b> and defaults to five.
<b>touch</b> <i>msg_lst</i>	When operating with your system mailbox, this command marks the messages in <i>msg_lst</i> to be moved to your personal mailbox when you quit the <b>mail</b> program, even though you have not read the listed messages. The messages appear in your personal mailbox as unread messages. When you use <b>touch</b> , the last message in <i>msg_lst</i> becomes the current message.

Figure 3 (Part 5 of 6). Mailbox Commands

## mail, Mail

---

Command	Function
<b>type</b> <i>msg_lst</i>	(t) Identical to the <b>print</b> command.
<b>Type</b> <i>msg_lst</i>	(T) Identical to the <b>Print</b> command.
<b>unalias</b> <i>al_lst</i>	Removes the defined aliases specified in <i>al_lst</i> .
<b>undelete</b> <i>msg_lst</i>	(u) Removes the messages in <i>msg_lst</i> from the list of messages to be deleted when you quit <b>mail</b> .
<b>unread</b> <i>msg_lst</i>	(U) Marks each message in <i>msg_lst</i> as <i>not</i> having been read.
<b>Unread</b> <i>msg_lst</i>	Identical to the <b>unread</b> command.
<b>unset</b> <i>option_lst</i>	Discards the values of the options specified in <i>option_lst</i> . This action is the inverse of the <b>set</b> command.
<b>version</b>	(ve) Displays the version banner for the <b>mail</b> program.
<b>visual</b> <i>msg</i>	(v) Activates the editor that you define with the <code>set VISUAL=</code> statement and loads message <i>msg</i> into the editor. When you exit the editor, the saved message is replaced in the mailbox being processed.
<b>write</b> <i>msg_lst file</i>	(w) Appends the messages specified in <i>msg_lst</i> to <i>file</i> . Displays the file name and the size of the file when the operation is complete. Does not include message headers in the file.
<b>xit</b>	(x) Identical to the <b>exit</b> command.
<b>z</b> [+ ] [-]	Changes the current message group (group of 20 messages) and displays the headings of the messages in that group. If a + or no argument is give, then headings in the next group are shown. If a - argument is given, the headings in the previous group are shown.

Figure 3 (Part 6 of 6). Mailbox Commands

## Handling Outgoing Mail

To compose and send a message, use **mail** with the following format:

```
mail addr_lst
```

In this format *addr\_lst* is a list of user addresses separated by spaces. This command activates the mail editor so that you can compose a message to be sent to the specified addresses.

By default, **mail** treats lines beginning with the character ~ (tilde) as special while you are composing a message. For instance, typing ~**m** on a line by itself places a copy of the current message into the response, shifting it to the right by one tab stop.

Other escapes set up subject fields, add and delete recipients of the message, and allow the user to escape to an editor to revise the message, or to a shell to run other commands. You can change the escape character to something other than a tilde with the **set escape = statement**. To view a summary of many useful commands, enter ~? on a line by itself while in the mail editor.

Figure 4 shows a summary of the mail editor commands. Use these commands only while in the mail editor. The editor recognizes commands only if you enter them at the beginning of a new line.

Command	Function
~!cmd	Executes the shell command, <i>cmd</i> and returns to the message.
~b <i>addr_lst</i>	Adds names in <i>addr_lst</i> to the list of people to receive blind copies of the message.
~c <i>addr_lst</i>	Adds names in <i>addr_lst</i> to the list of people to receive copies of the message.
~d	Reads the file <b>dead.letter</b> from your home directory into the message.
~e	Activates the editor that you have specified with the <b>set EDITOR=</b> statement using the message text in the current message. When you exit that editor, you return to the mail editor to continue appending the changed message, or to send the message by exiting the <b>mail</b> program.
~f <i>msg_lst</i>	Reads the named messages into the message being sent. If no messages are specified, reads the current message. This command works only if you entered the mail editor from the mailbox listing using the <b>m</b> or <b>r</b> mailbox commands.
~h	Allows you to edit the message header fields by typing each one in turn. Allows you to append text to the end or modify the field using the current terminal erase and kill characters.

Figure 4 (Part 1 of 2). Mail Editor Commands

## mail, Mail

---

Command	Function
<code>~m msg_lst</code>	Reads the named messages into the message being sent, shifted right one tab. If no messages are specified, reads the current message. This command works only if you entered the mail editor from the mailbox listing using the <b>m</b> or <b>r</b> mailbox commands.
<code>~p</code>	Displays the message as it currently exists, prefaced by the message header fields.
<code>~q</code>	Aborts the message being created without sending it. Saves the message in <b>dead.letter</b> in your home directory if the <b>save</b> option is set.
<code>~r filename</code>	Reads the named file into the message.
<code>~s string</code>	Changes the <b>Subject:</b> field to the phrase specified in <i>string</i> .
<code>~t addr_lst</code>	Adds the addresses in <i>addr_lst</i> to the <b>To:</b> field of the message.
<code>~v</code>	Activates the editor that you have specified with the <code>set VISUAL=</code> statement using the message text in the current message. When you exit that editor, you return to the mail editor to continue appending to the changed message, or to send the message by exiting the <b>mail</b> program.
<code>~w filename</code>	Writes the message to the named file.
<code>~!cmd</code>	Pipes the message through the command <i>cmd</i> as a filter. If <i>cmd</i> gives no output or terminates abnormally, it retains the original text of the message. Otherwise, the output of <i>cmd</i> replaces the current message. The command <b>fmt</b> is often used as <i>command</i> to rejustify the message.
<code>~~</code>	Allows you to use the character <code>~</code> (tilde) in a message without it being interpreted as a command prefix. The sequence <code>~~</code> results in only one <code>~</code> being sent in the message. If you have changed the escape character, double that character instead of <code>~</code> to use the new escape character as a single character.

**Figure 4 (Part 2 of 2). Mail Editor Commands**

You can end a **mail** session with the **quit (q)** command. Messages that you have looked at go to your personal mailbox. Messages that you have marked to be deleted are deleted. Messages that you have not looked at go back to your system mailbox.

## Customizing the Mail Program

The **mail** command has a number of options that you can set to customize the mail system for your particular use. Use the **set** command to enable options, and the **unset** command to disable options. You can also use the **set** command to assign a value to an option.

The format for using the **set** command to enable options is:

```
set [option-list]
```

The *option-list* may be one or more options that you want to enable. To set options so that they are valid each time you use **mail**, put the commands in **.mailrc** in your \$HOME directory. To set options so that they are valid for all users on the system, put the commands in **/usr/lib/Mail.rc**. The following table, Figure 5, lists the binary options (those that need only be set or unset).

Option	Function
<b>append</b>	Causes messages saved in <b>mbox</b> to be appended (added to the end) rather than prepended (added to the beginning).
<b>ask</b>	Causes <b>mail</b> to prompt you for the subject of each message you send. If you respond with a new line (carriage return), no subject field is set.
<b>askcc</b>	Causes you to be prompted for the addresses of people to receive copies of the message. Responding with a new line indicates your satisfaction with the current list.
<b>autoprint</b>	Causes the <b>delete</b> command to behave like <b>dp</b> . Thus, after deleting a message, the next one is typed automatically.
<b>debug</b>	Same as specifying <b>-d</b> on the command line. Causes <b>mail</b> to display debugging information. <b>mail</b> does not send mail while in debug mode.
<b>dot</b>	Causes <b>mail</b> to interpret a period alone on a line as the terminator of a message you are sending.
<b>hold</b>	Holds messages in the system mailbox by default.
<b>ignore</b>	Causes interrupt signals from your terminal to be ignored and echoed as <b>@'s</b> .

Figure 5 (Part 1 of 2). Binary Options

## mail, Mail

---

Option	Function
<b>ignoreeof</b>	Related to <b>dot</b> . Makes <b>mail</b> refuse to accept an <b>Ctrl-D</b> as the end of a message. <b>ignoreeof</b> also applies to <b>mail</b> command mode.
<b>metoo</b>	Usually, when an alias containing the sender is expanded, the sender is removed from the expansion. Setting this option causes the sender to be included in the alias expansion (and thus receives copies of messages).
<b>nosave</b>	Normally, when a message is terminated with two interrupt sequences ( <b>Alt-Pause</b> ), <b>mail</b> copies the partial letter to the file <b>dead.letter</b> in your home directory. Setting the binary option <b>nosave</b> prevents this.
<b>Replyall</b>	Reverses the sense of the <b>reply</b> and <b>Reply</b> mailbox commands.
<b>quiet</b>	Suppresses the printing of the program banner when <b>mail</b> starts.
<b>verbose</b>	Same as using the <b>-v</b> flag on the command line. When <b>mail</b> runs in verbose mode, the actual delivery of messages is displayed on the user's terminal.

Figure 5 (Part 2 of 2). Binary Options

The following table, Figure 6 lists the valued options (those that need to be assigned a value).

Option	Function
<b>EDITOR</b>	Path name of the text editor to use in the <b>edit</b> command and <b>~e</b> escape. If not defined, then a default editor ( <b>/usr/bin/e</b> ) is used.
<b>PAGER</b>	Path name of the paging program to use for the <b>more</b> command or when the <b>crt</b> variable is set. If you do not specify a value for <b>PAGER</b> , the system uses <b>/bin/pg</b> .
<b>SHELL</b>	Path name of the shell to use in the <b>!</b> command and the <b>~!</b> escape. A default shell is used if this option is not defined.
<b>VISUAL</b>	Path name of the text editor to use in the <b>visual</b> command and <b>~v</b> escape. The default path name is <b>/usr/bin/vi</b> .

Figure 6 (Part 1 of 2). Valued Options

Option	Function
<b>crt = <i>n</i></b>	Calls the <b>pg</b> command to display the message when the message exceeds <i>n</i> lines.
<b>escape</b>	If defined, the first character of this option gives the character to use in the place of ~ to denote escapes.
<b>folder</b>	Defines the name of the directory to use for storing folders of messages. If this name begins with a /, <b>mail</b> considers it to be an absolute path name; otherwise, the folder directory is found relative to your home directory.
<b>record</b>	If defined, gives the path name of the file (relative to \$HOME) used to record all outgoing mail. If not defined, then outgoing mail is not saved. Do not include the home directory as part of the path name.
<b>screen</b>	If defined, controls the size of the window for message headers. You can set this option to show the number of lines on the screen. For example, the entry screen=22 causes the system to scroll for 22 lines and then pause.
<b>toplines</b>	If defined, gives the number of lines of a message to be printed out with the <b>top</b> command; normally, the first five lines are printed.

Figure 6 (Part 2 of 2). Valued Options

## Flags

<b>-v</b>	Puts <b>mail</b> into verbose mode. Details of delivery are displayed on the user's terminal.
<b>-i</b>	Causes tty interrupt signals to be ignored. Useful when using <b>mail</b> on noisy phone lines.
<b>-n</b>	Inhibits the reading the <b>/usr/lib/Mail.rc</b> .
<b>-N</b>	Suppresses the initial printing of headers.
<b>-s <i>subject</i></b>	Specifies a subject for a message to be created.
<b>-f <i>name</i></b>	Causes <b>mail</b> to read in the contents of your <b>mbox</b> or the specified file for processing. When you <b>quit</b> , <b>mail</b> writes undeleted messages back to this file.
<b>-u <i>user-id</i></b>	Short way of doing <b>mail -f /usr/mail/<i>user-id</i></b> . Activates <b>mail</b> for a specified user's mailbox. You must have access permission to the specified mailbox.

## mail, Mail

---

### Files

<code>/usr/mail/*</code>	System mailboxes for all users.
<code>\$HOME/mbox</code>	Your personal mailbox.
<code>\$HOME/.mailrc</code>	File containing mail commands to customize <b>mail</b> to a specific user.
<code>/tmp/R#</code>	Temporary for editor escape.
<code>/usr/lib/Mail.help</code>	Help file for mailbox commands.
<code>/usr/lib/Mail.tildehelp</code>	Help file for mail editor commands.
<code>/usr/lib/Mail.rc</code>	File containing mail commands to change <b>mail</b> for all users on the system.

### Related Information

The following commands: “**bellmail**” on page 104, “**sendmail**” on page 897, and “**uucp**” on page 1144.

The chapter about sending and receiving mail in *IBM RT Using the AIX Operating System*.

The chapters about Mail Handler (an alternative to **mail**) and about managing the mail system in *IBM RT Managing the AIX Operating System*.

---

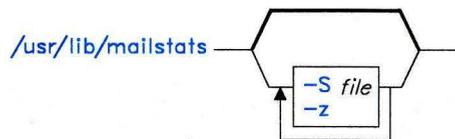
## mailstats

---

### Purpose

Displays statistics regarding mail traffic.

### Syntax



AJ2FL146

### Description

This command reads the information in `/usr/adm/sendmail/sendmail.st` (or in the file specified with the `-S` flag), formats it, and writes it to standard output. The format of the information is shown in the following example:

```

Sendmail statistics from file "/usr/adm/sendmail/sendmail.st"
Collection started at Thu Feb 18 17:40:41 1988
  
```

Mailer	msgs_from	bytes_from	msgs_to	bytes_to
-----	-----	-----	-----	-----
local	1	2	1	201
prog	0	0	0	0
uucp	0	0	0	0
tcp	0	0	0	0

The fields in the report have the following meanings:

**Mailer** This field contains the name of the mailer program that handled the mail.

**msgs\_from**

This field (*messages from*) contains the number of messages that originated from the indicated mailer.

**bytes\_from**

This field contains the number of bytes of information in the messages sent from the indicated mailer.

## mailstats

---

**msgs\_to** This field (*messages to*) contains the number of messages that ended locally and were received by the indicated mailer.

**bytes\_to** This field contains the number of bytes of information in the messages received by the indicated mailer.

The collection start time indicated on the second line of the report is the time at which the first update to the empty file was performed.

If **sendmail** transmits mail directly to a file, such as **dead.letter** or an alias target, the message and byte counts are credited to the **prog** mailer in addition to the normal statistics for use of the **prog** mailer.

## Statistics Messages

When **mailstats** is called with no program flags, it can generate the following messages:

No statistics data in file "/usr/adm/sendmail/sendmail.st"

The **sendmail** program has not written any data into the statistics file.

mailstats: file size change; use previous mailstats version

The statistics file format is not the format expected by **mailstats**. Try using a previous version of **mailstats** to read it.

## Flags

**-S file** Specifies to use *file* as the input statistics file instead of **/usr/adm/sendmail/sendmail.st**

**-z** Clears the contents of the statistics file. Clearing the file erases the contents and allows you to start gathering statistics again.

## Files

/usr/lib/mailstats	The <b>mailstats</b> program.
/usr/adm/sendmail/sendmail.st	The database file containing mail system statistics.
/usr/adm/sendmail/sendmail.cf	The configuration file for <b>sendmail</b> program.

## Related Information

The command: “**sendmail**” on page 897.

The chapter about managing the mail system in *IBM RT Managing the AIX Operating System*.

---

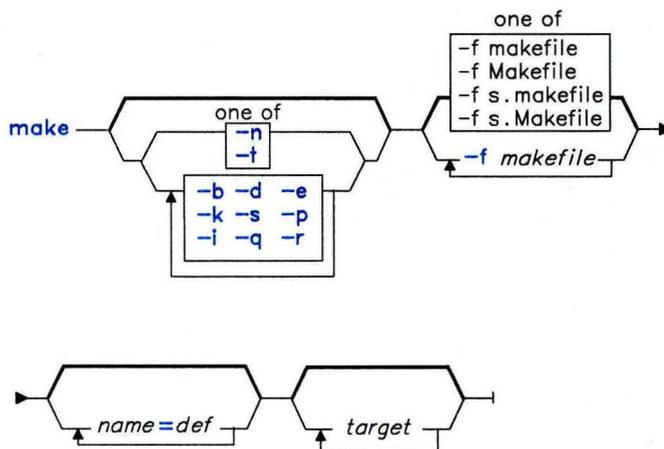
# make

---

## Purpose

Maintains up-to-date versions of programs.

## Syntax



OL805035

## Description

The **make** command reads *makefile* for information about the specified *target* files and for the commands necessary to update them. **make** does not change the *target* if you have not changed any of the source files since you last built it. It considers a missing file to be a changed file (out-of-date).

You can also include macro definitions on the command line after all of the flags. Macro definitions have the form:

*macro-name* = *string*

See “Macros” on page 628 for more information about macros and their uses.

The **make** command considers all entries on the command line that follow the flags and that do not contain an equal sign to be target file names.

## Description File

The description file contains a sequence of entries specifying the files that the target files depend on. The general form of an entry is:

```
targ [targ] . . . :[:]file] . . . [cmd] . . . [#]  
                                [cmd] . . . [#]
```

The first line of an entry (called the **dependency line**), contains a list of targets followed by a : (colon) and an optional list of prerequisite files or dependencies. If you put shell commands on the dependency line, they must be preceded by a ; (semicolon). All commands that follow the semicolon and all following lines that begin with a tab contain shell commands that **make** uses to build the target.

To specify more than one set of commands, you must enter more than one dependency definition. In this case, each definition must have the target name followed by two colons (::), a dependency list, and a command list.

The first line that does not begin with a tab or # (pound sign) begins a new dependency or a macro definition. Command lines are performed one at time, each by its own subshell. Thus, the effect of some shell commands, such as **cd**, does not extend across new-line characters. You can, however, put a \ (backslash) at the end of a line to continue it on the next physical line. A comment begins with a # and ends with a new-line character.

The first one or two characters in a command can be one of the following special characters:

- Ignores errors returned by the command on this line.
- @: Does not display this command line.
- @
- @- Does not display this command line and ignores errors.

## Suffixes

The **make** command has default rules that govern the building of most standard files. These rules depend on the standard suffixes used by the system utility programs to identify file types. These rules define the starting and ending file types so that, for example, given a specified **.o** file, **make** can infer the existence of a corresponding **.c** file and knows to compile it using the **CC -C** command.

A rule with only one suffix (that is, **.c:**) defines the building of *prog* from all its source files. Use a ~ (tilde) in the suffix to indicate a SCCS file. For example, the **.c~.o** rule governs changing an SCCS C source file into an object file. You can define rules within the description file. **make** recognizes as a rule any target that contains no slashes and starts with a dot.

You can also add suffixes to the list of suffixes recognized by **make** and add to the default dependency rules. Use the target name **.SUFFIXES** followed by the suffixes you want to add. Be careful of the order in which you list the suffixes. **make** uses the first possible name for which both a file and a rule exist. The default list is:

```
.SUFFIXES: .o .c .c~ .y
.y~ .l .l~ .s .s~
.sh .sh~ .h .h~
```

You can clear the list of suffixes by including **.SUFFIXES:** with no following list.

## Special Target Names

You can use some special target names in the description file to tell **make** to process the file in a different manner. The special target names are:

- .DEFAULT**      The commands that appear after this name in the description file tell **make** what to do if it can find no commands or default rules to tell it how to create a specific file.
- .IGNORE**        If this name appears on a line by itself, **make** does not stop when errors occur. Using a - (minus) as the first character on a line in the description file tells **make** to ignore errors for the command on that line.
- .PRECIOUS**     The files named on the same line as this special name are not removed when **make** is interrupted.
- .SILENT**        If this name appears on a line by itself, **make** does not display any of the commands that it performs to build a file.
- .SUFFIXES**      Use this name to add more suffixes to the list of file suffixes that **make** recognizes.

## Environment

When you run **make**, it reads the environment and treats all variables as macro definitions. **make** processes the environment variables after it processes its own internal rules and before processing any description files. Therefore, macro assignments in a description file normally override duplicate environment variables. The **-e** flag instructs **make** to use the environment variables instead of the description file macro assignments.

The **make** command recognizes a macro **MAKEFLAGS**, which can be assigned any **make** command line flag except **-f**, **-p**, and **-d**. When **make** begins, it assigns the current flags to **MAKEFLAGS**. It passes this variable to any commands it invokes, including additional invocations of **make** itself. Thus you can perform a **make -n** recursively on a software system to see what would have been performed. The **-n** is put in **MAKEFLAGS** and passed to further copies of the shell that runs the next level of **make** commands. In this way, you can check all of the description files for a software project without actually compiling the project.

## make

---

**Note:** Some **makefile** macros can conflict with **csh** variable substitutions. You should avoid using **make** with the **csh** shell. The **sh** shell does not conflict with **make** macros and it is the recommended shell. Otherwise, you can avoid conflicts by adding a `SHELL = /bin/sh` to the makefile.

### Macros

Entries of the form `string1 = string2` are macro definitions. `string2` can consist of all characters that can occur on a line before a comment character (`#`) or before a new-line character that is not a continuation line. After this macro definition, **make** replaces each `$(string1)` in the file with `string2`. You do not have to use the parentheses around the macro name if the macro name is only one character long and there is no substitute sequence (see the next paragraph). If you use the following form, you can also replace characters in the macro string with other characters for one time that you use the macro:

```
$(string1[:subst1=[subst2]])
```

The optional `:subst1 = subst2` + specifies a substitute sequence. If you specify a substitute sequence, **make** replaces each `subst1` in the named macro with `subst2` (if `subst1` does not overlap with another `subst1`). Strings in a substitute sequence begin and end with any of the following: a blank, tab, new-line character, or beginning of line. See “Libraries” on page 629 for an example of the use of the substitute sequence.

**Note:** Because **make** uses the dollar sign symbol (`$`) to designate a macro, do not use that symbol in file names of targets and parents, or in commands in the description file unless you are using a defined **make** macro.

### Internal Macros

The **make** command has five internal macros. It assigns values to these macros under one or more of the following conditions:

- When it uses an internal rule to build a file.
- When it uses a **.DEFAULT** rule to build a file.
- When it uses rules in the description file to build a file.
- When the file is a library member.

They are defined as follows:

`$*` The file name (without the suffix) of the source file.

`$$` The full target name of the current target.

`$$` The source files of an out-of-date module. **make** evaluates this macro when applying inference rules or the **.DEFAULT** rule. For example:

```
.C.O:  
cc -c $$
```

Here,  $\$<$  is the equivalent of  $\$*$  and refers to the `.c` file of any out-of-date `.o` file.

$\$?$  The list of out-of-date files. **make** evaluates this macro when it evaluates explicit rules from *makefile*.

$\$%$  The name of an archive library member. **make** evaluates this macro only if the target is an archive library member of the form *lib(file.o)*. In this case,  $\$@$  evaluates to *lib* and  $\$%$  evaluates to the library member, *file.o*.

You can add an uppercase **D** or **F** to indicate “directory part” or “file part,” respectively, to all internal macros except for  $\$?$ . Thus,  $\$(@D)$  refers to the directory part of the name  $\$@$ . If there is no directory part, **make** uses `./`.

## Libraries

If a target name contains parentheses, **make** considers it an archive library. The string within parentheses refers to a library member. Thus, *lib(file.o)* and  $\$(LIB)(file.o)$  both see an archive library which contains *file.o*. (You must have already defined the **LIB** macro.) The expression  $\$(LIB)(file1.o file2.o)$  is not legal.

Rules that apply to archive libraries have the form *x.a*, where *x* is the suffix of the file you want to add to an archive library. For example, `.C.a` indicates a rule that changes any C source file to a library file member. The following lines give the default rule for this change:

```
lib: lib(file1.o) lib(file2.o) lib(file3.o)
    @echo lib is now up to date
.c.a:
    $(CC) -c $(CFLAGS) $<
    ar rv $@ $*.o
    rm -f $*.o
```

*x* must be different from the suffix of the archive member. Therefore, you cannot have *lib(file.o)* depend upon *file.o*.

Another, but more limited, example of an archive library maintenance rule follows:

```
lib: lib(file1.o) lib(file2.o) lib(file3.o)
    $(CC) -c $(CFLAGS) $(:.o=.c)
    ar rv lib $?
    rm $? @echo lib is now up to date
.c.a:;
```

This example rule uses a substitute sequence (`.o=.c`) to replace with `.c` files all `.o` files generated by the  $\$?$  macro. The  $\$?$  list is the set of object file names (inside *lib*) with C source files that are out of date. The macro substitution translates `.o` to `.c`.

## make

---

If this rule appears in your description file, it disables the default `.c.a:` rule, which creates each object file one by one. This type of organization speeds up archive library maintenance, but becomes hard to use if the archive library contains a mix of assembly programs and C programs.

### Flags

- b** Recognizes *makefiles* that were written for old versions of **make**.
- d** Displays detailed information about the files and times that **make** examines (debug mode).
- e** Uses environment variables in place of any assignments made within description files. These assignments normally replace environment variables.
- f *makefile*** Reads *makefile* for a description of how to build the target file. If you give only a - (minus) for *makefile*, **make** reads standard input. If you do not use the **-f** flag, **make** looks in the current directory for a description file named **makefile**, **Makefile**, **s.makefile**, or **s.Makefile**. You can specify more than one description file by entering the **-f** flag more than once (with its associated *makefile* parameter).
- i** Ignores error codes returned by commands. **make** normally stops if a command returns a nonzero code. Use this flag to compile several modules only if you want **make** to continue when an error occurs in one of the modules. Do not link the resulting modules when you use this flag.
- k** Stops processing the current target if an error occurs, but continues with other branches that do not depend on that target.
- n** Displays commands, but do not run them. Displays lines beginning with an @ (at sign). If the command in the description file contains the string `$(MAKE)`, perform another call to **make** (see the discussion of the **MAKEFLAGS** macro on page 627). Use this flag to preview the performance of **make**.
- p** Displays the complete set of macro definitions and target descriptions before performing any commands.
- q** Returns a zero status code if the target file is up to date; returns a nonzero status code if the target file is not up to date.
- r** Does not use the default rules.
- s** Does not display commands on the screen as they are performed.

- t** Changes only the date of the files, rather than performing the listed commands. Use this flag if you have made only minor changes to a source file that do not affect anything outside of that file. This flag changes the date of all target files that appear on the command line or in the description file.

## Examples

1. To make the file specified by the first entry in the description file:

```
make
```

2. To display, but not run, the commands that **make** would use to make a file:

```
make -n search.o
```

You may want to do this to verify that a new description file is correct before using it.

3. To save the internal rules in a file:

```
make -p -f /dev/null 2> /dev/null > defaults
```

This lists the internal rules and macros and saves them in the file `defaults` for viewing or editing. All exported shell environment variables are included in the list of macro definitions.

## Files

Makefile  
makefile  
s.Makefile  
s.makefile

## Related Information

The discussion of **make** in *AIX Operating System Programming Tools and Interfaces*.

# makedbm

---

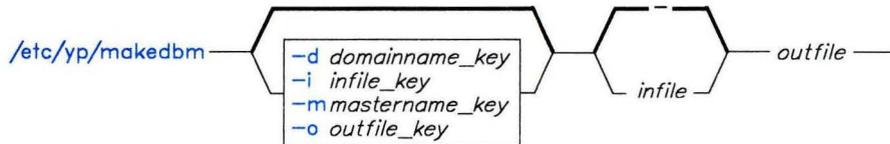
## makedbm

---

### Purpose

Makes a Yellow Pages **dbm** map.

### Syntax



A5AC5004

### Description

The **makedbm** command is most often invoked from **/etc/yp/Makefile** to generate Yellow Pages maps. The **makedbm** command converts *infile* to a pair of files in **dbm** format. The two files are **outfile.pag** and **outfile.dir**. Each line in the input file is converted to a single **dbm** record. All characters up to the first space or tab form the key while the rest of the line is the value data. If a line ends with \ (backslash), data for that record is continued on the next line. Yellow Pages clients must interpret the # symbol since **makedbm** does not treat it as a comment character. If *infile* is - (minus sign), **makedbm** reads standard input.

The **makedbm** command generates a special entry with the key **YP\_LAST\_MODIFIED** giving the date of *infile* or the current time if *infile* is specified to be standard input.

---

#### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

### Flags

- d Creates a special entry with the key **YP\_DOMAIN\_NAME**.
- i Creates a special entry with the key **YP\_INPUT\_FILE**.
- o Creates a special entry with the key **YP\_OUTPUT\_NAME**.

- m** Creates a special entry with the key *YP\_MASTER\_NAME*. If no master host name is specified, *YP\_MASTER\_NAME* is set to the local host name.
- u** Displays individual entries in a **dbm** file, with a single space separating keys from values.

### Example

In the following example, data from the `/etc/passwd` file is converted to a form that **makedbm** can use to create the Yellow Pages map **passwd.nam**:

```
awk '{FS=":"; OFS="\t"; print $1,$0}' \  
etc/passwd | makedbm - passwd.nam
```

### File

`/etc/yp/Makefile`

### Related Information

The following commands: “**yppush**” on page 1252 and “**ypinit**” on page 1243.

# makekey

---

## makekey

---

### Purpose

Generates an encryption key.

### Syntax

```
/usr/lib/makekey —
```

OL805240

### Description

The **makekey** command generates an encryption key to use with programs that perform encryption. Its input and output are usually pipes.

The **makekey** command reads 10 characters from standard input and writes 13 characters to standard output. The first 8 of the 10 input characters can be any sequence of ASCII characters. The last two input characters (the *salt*), are best chosen from the set [a-zA-Z0-9.,/]. The salt characters are repeated as the first two characters of the output. The remaining 11 output characters are chosen from the same set as the salt and constitute the output key that you use as the *key* parameter to programs that perform encryption.

---

#### Japanese Language Support Information

This command has not been modified to support Japanese characters.

---

### Example

To generate an encryption key:

```
/usr/lib/makekey  
1234567890
```

This generates an encryption key based on the string 1234567890. The key 90y74T/NXw1U is displayed at the work station. Do *not* press **Ctrl-D** after typing the input key 1234567890 because this would end your shell session. Also, the shell prompt appears immediately after the generated key, instead of appearing on a separate line as it usually does. This is normal.

---

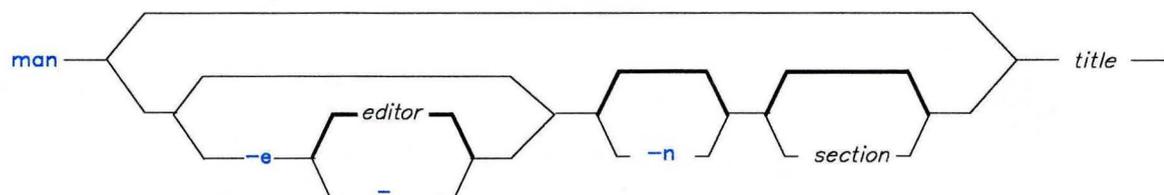
## man

---

### Purpose

Displays manual entries online.

### Syntax



AJ2FL103

### Description

The **man** command locates and displays the entries for the specified *title* and *section* of the online manual. If specified, **man** displays entries through *editor*. If an *editor* is not specified, **man** displays entries by default through the editor set by the **EDITOR** environment variable. If the **EDITOR** environment variable is not set, then the entry is displayed on standard output.

The section number may be one of the following:

- Commands and Application Programs
- System Calls
- Subroutines
- Special Files
- File Formats
- Games
- Miscellaneous Facilities.

If no section is specified, then all sections are searched for each title and all such occurrences are displayed; if a given title is not found in the specified section, then all sections are searched for that title.

**Note:** This command is available optionally; it is not part of the standard AIX Operating System package.

## man

---

### Flag

- `-e editor` Displays entries with the specified *editor*.
- `-e -` Does not display entries with the *editor*.
- `-n` Displays entries without pagination.

### Files

```
/usr/bin/man  
/usr/man/cat[1-7]/*  
/usr/bin/man[i l n o p]
```

### Examples

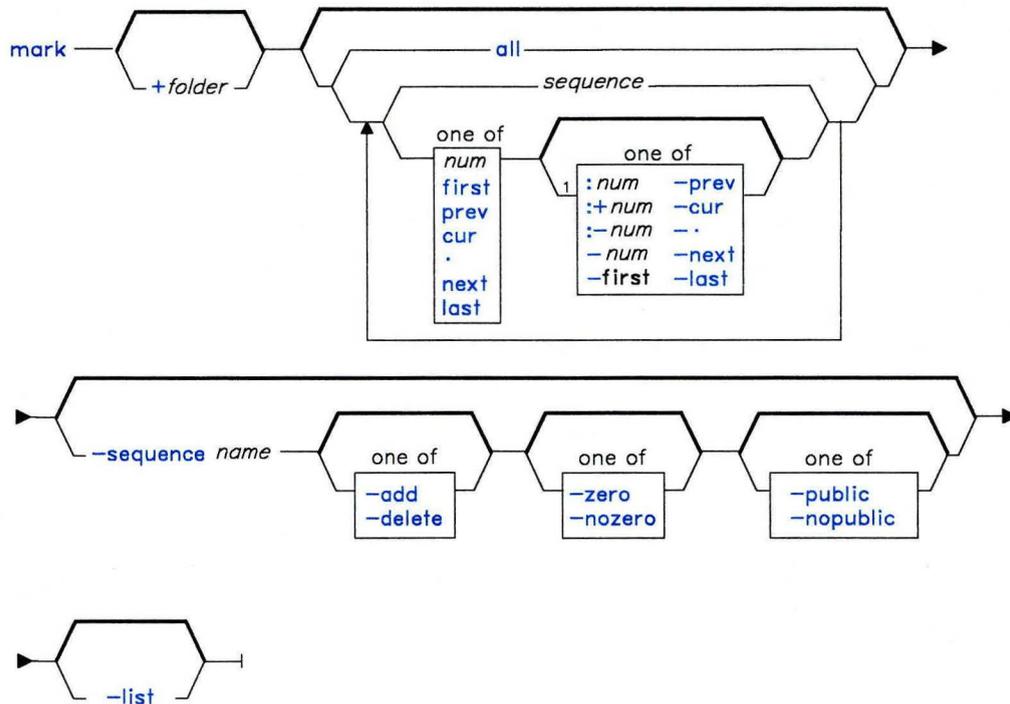
1. To display the **man** entry on standard output when the **EDITOR** variable is not set:  
`man man`
2. To view section 1 of this entry in the **vi** editor:  
`man -e vi 1 man`

# mark

## Purpose

Creates, modifies, and displays message sequences.

## Syntax



AJ2FL200

mark — -help —|

AJ2FL201

<sup>1</sup> Do not put a blank between these items.

OL805308

# mark

---

## Description

The **mark** command is used to create a sequence, delete a sequence, add messages to a sequence, and delete messages from a sequence. The **mark** command is also used to list messages in a sequence and list sequences in a folder. The **mark** command is part of the MH (Message Handling) package and can be used with other MH and AIX commands.

The **mark** command specified with only a folder name lists the sequences defined for that folder and the messages that comprise each of the sequences. If you specify a new sequence name with the **-sequence** flag, **mark** creates a new sequence. You can use the **-add** flag to add messages to a sequence and the **-delete** flag to remove messages from a sequence. When all messages are deleted from a sequence, **mark** removes the sequence name from the folder.

## Flags

- add** Adds messages to the specified sequence. If messages are specified, **-add** is the default.
- delete** Deletes messages from the specified sequence.
- + folder msgs** Specifies messages that you want **mark** to select from. *msgs* can be several messages, a range of messages, or a single message. You can use the following message references when specifying *msgs*:
- |             |              |                 |
|-------------|--------------|-----------------|
| <i>num</i>  | <b>first</b> | <b>prev</b>     |
| <b>cur</b>  | <b>.</b>     | <b>next</b>     |
| <b>last</b> | <b>all</b>   | <i>sequence</i> |
- If the **-list** flag is used, the default for *msgs* is **all**. Otherwise, the default is the current message. The default folder is the current folder. If you specify a folder, that folder becomes the current folder.
- help** Displays help information for the command.
- list** Displays the messages in the specified sequence. If you do not specify a sequence, **-list** displays all sequence names defined for the folder and the messages in each sequence.
- npublic** Restricts the specified sequence to your usage. **-npublic** does not restrict the messages in the sequence, only the sequence. This option is the default if the folder is write-protected from other users.
- nozero** Modifies the sequence by adding or deleting only the specified messages (see the **-zero** flag). This flag is the default.
- public** Makes the specified sequence available to other users. **-public** does not make protected messages available, only the sequence. This flag is the default if the folder is not write-protected from other users.

- sequence *name*** Specifies a sequence for the **-list**, **-add**, and **-delete** operations. You must specify this flag for the **-add** and **-delete** operations.
- zero** Clears the specified sequence of all messages before adding any other messages. When the **-delete** flag is also specified, **-zero** places all of the messages from the folder into the sequence before deleting any messages.

## Profile Entries

- Current-Folder:** Sets your default current folder.  
**Path:** Specifies your *user\_mh\_directory*.

## Files

- `$HOME/.mh_profile` The MH user profile.

## Related Information

The MH command “**pick**” on page 748.

The **mh-profile** file in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

# mdrc

---

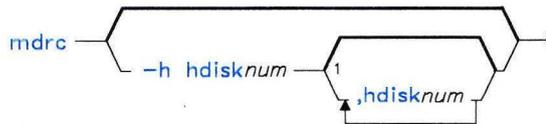
## mdrc

---

### Purpose

Allows you to reinstall a user-created minidisk after you have reinstalled AIX.

### Syntax



OL805440

### Description

The **mdrc** command provides access to user-created minidisks. You should run this command if you have reinstalled the AIX Operating System or if you have had to replace the **/etc/system**, **/etc/filesystems**, or **/etc/ddi/cpmgr** files with copies that do not contain stanzas describing any user-installed minidisks. The system uses the information in these stanzas to configure the minidisks at system startup, and **mdrc** recreates the necessary stanzas. Normally, **mdrc** uses the backup copy of **/etc/filesystems** produced by the **minidisks** command when you use it to create a new minidisk. This backup copy is named **/u/filesystems**.

If **mdrc** cannot recreate the original **/etc/filesystems** stanza for AIX Operating System minidisks, it assigns attributes of **Auto Mount=no**, **Read/Write Status=R/W**, and **Mount Directory=/tmp/directory/hdn** to the minidisk. In this case, you should then run the **minidisk** command to change the attributes to the values you want. You might also need to run the **mkdir** command to create the mount directory, if you reinstalled the entire AIX Operating System.

If a minidisk has been created for use by the Personal Computer AT Coprocessor<sup>4</sup>, **mdrc** will update the **/etc/ddi/cpmgr** file. If you have not installed Personal Computer AT Coprocessor Services before running **mdrc**, it creates an entry in **/etc/system**, but displays a warning message because the **/etc/ddi/cpmgr** does not exist. You must run **mdrc** again after you install the Coprocessor to be able to use the coprocessor minidisks.

---

<sup>4</sup> Personal Computer AT Coprocessor is a registered trademark of International Business Machines Corporation.

The **mdrc** command does not recognize external disks, or any minidisks on them, if the disks are not configured. To configure an external disk and its minidisks, see “**varyon**” on page 1180.

You must have superuser authority or be a member of the system group to run the **mdrc** command. When auditing is on, an audit record of the type **mdrc** is created.

## Flag

**-h hdisknum[, hdisknum] . . .**

Specifies any disks that have been removed or damaged and tells **mdrc** to remove the minidisk configuration entries for these disks. If you do not specify this flag and an external disk is not configured, **mdrc** ignores entries in the configuration files for the external disk’s minidisks.

**Note:** If you do not have any external disks, you do not need to specify this flag.

## Files

/etc/filesystems  
/etc/system  
/etc/ddi/cpmgr

## Related Information

The following commands: “**watch**” on page 1209, “**mkdir**” on page 657, and “**varyon**” on page 1180.

The **filesystems** and **system** files in *AIX Operating System Technical Reference*.

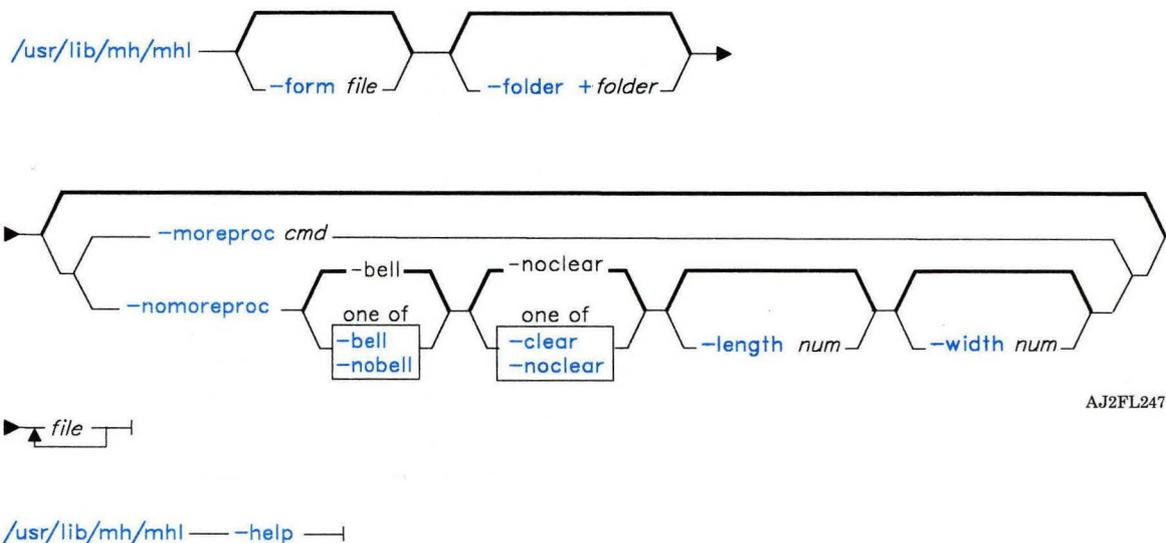


# mhl

## Purpose

Produces formatted listings of messages.

## Syntax



## Description

The **mhl** command is used to create formatted lists of messages. The **mhl** command is part of the MH (Message Handling) package and can be used with other MH and AIX commands. **mhl** is usually invoked through the profile entry **showproc:** or through the **-showproc** flag in other MH commands.

The **mhl** command uses the formatting directions listed in the format file to display the message information about each of the specified messages. If you specify more than one message, **mhl** provides a prompt before displaying each screen of messages or with **-nomoreproc**, before displaying each message. If you specify **-nomoreproc**, press **Enter** or **END OF FILE** to see the next message. Press **INTERRUPT** to stop the current message output and to receive a prompt for the next message. Press **QUIT** to stop the command output.

# mhl

---

## Flags

- bell** Produces an audible indicator at the end of each page. This flag affects **mhl** only if the output device is a display and the **moreproc:** entry is defined and empty. This flag is the default.
- clear** Clears the screen at the end of each page when the output device is a display. When the output device is not a display, **-clear** inserts a form feed character at the end of each message. If the output device is a display, **mhl** uses the **\$TERM** and **\$TERMCAP** environment variables to determine the type of display. This flag affects **mhl** only if the **moreproc:** entry is defined and is empty.
- folder +folder** Specifies the folder to be used for the **mhl.format** *messagename* entry. The default is the value of the **\$mhfolder** environment variable.
- form file** Uses the format contained in the specified file. If you do not specify this flag, **mhl** uses the format described in *user-mh-directory/mhl.format*. If this file does not exist, **mhl** uses the system default format described in */usr/lib/mh/mhl.format*.
- help** Displays help information for the command.
- length num** Sets the length of the output. The default value is the value indicated by **\$TERMCAP**. If that value is not appropriate, the default value is 40.
- moreproc cmd** Uses *cmd* instead of the value of the **moreproc:** entry specified in **\$HOME/.mh-profile**.
- nobell** Does not produce an audible indicator at the end of each page. This flag affects **mhl** only if the output device is a display and the **moreproc:** entry is defined and is empty.
- noclear** Does not clear the screen at the end of each page when the output device is a display. When the output device is not a display, **-clear** does not insert a form feed character at the end of each message. This flag affects **mhl** only if the **moreproc:** entry is defined and is empty. This flag is the default.
- nomoreproc** Sets **moreproc:** as an empty value.
- width num** Sets the width of the output. The default value is the value indicated by **\$TERMCAP**. If that value is not appropriate, the default value is 80.

## Profile Entry

**moreproc:** Specifies the interactive program for communicating with user.

## Files

<code>/usr/lib/mh/mhl.format</code>	The default MH message template.
<code>user-mh-directory/mhl.format</code>	The user's default message template. (If it exists, it overrides the default MH message template.)
<code>\$HOME/.mh-profile</code>	The MH user profile.

## Related Information

Other MH commands: “**ap**” on page 53, “**dp**” on page 352, “**next**” on page 694, “**prev**” on page 765, and “**show**” on page 942.

The **mh-format** and **mh-profile** files in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

# mhmail

---

## mhmail

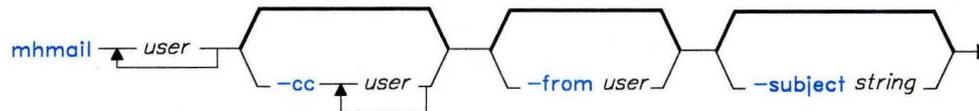
---

### Purpose

Sends or receives mail.

### Syntax

mhmail —|



mhmail — -help —|

AJ2FL236

### Description

The **mhmail** command is used to incorporate messages and compose messages. The **mhmail** command is part of the MH (Message Handling) package and can be used with MH and AIX commands.

The **mhmail** command entered by itself incorporates messages from your mailbox. If you specify user addresses, **mhmail** accepts text from your terminal and composes a message. You can end the message text by pressing **END OF FILE**. **mhmail** sends a copy of the message to each specified address.

## Flags

- body** *string* Sends a message with *string* as the body. When you specify **-body**, **mhmail** does not accept text from the terminal.
- cc** *users* Sends a copy of the message to the specified users. **mhmail** puts the addresses in the **cc:** field.
- from** *user* Places the specified user address in the **From:** field of the message.
- help** Displays help information for the command.
- subject** *string* Places the specified text string in the **Subject:** field of the message.

## Files

`/usr/mail/$USER` The location of the mail drop.

## Related Information

Other MH commands: “**inc**” on page 518 and “**post**” on page 758.

The **mh-alias**, **mh-mail**, and **mh-profile** files in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

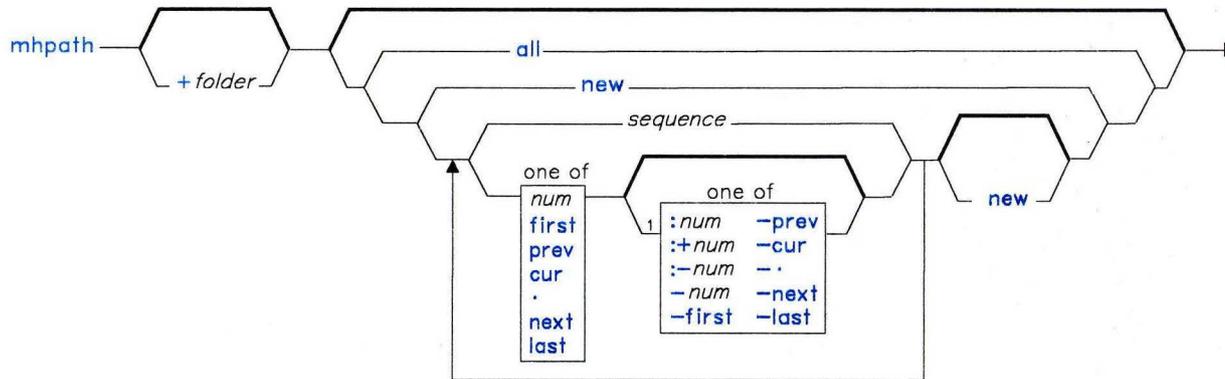
# mhpath

## mhpath

### Purpose

Prints full path names of messages and folders.

### Syntax



mhpah — -help —|

AJ2FL214

AJ2FL215

<sup>1</sup> Do not put a blank between these items.

OL805308

### Description

The **mhpah** command is used to list the path names of folders and messages. **mhpah** is part of the MH (Message Handling) package and can be used with other MH and AIX commands.

The **mhpah** command lists the path names of all specified messages. If you do not specify any messages, **mhpah** lists the path name of the folder. If you do not specify messages or a folder, **mhpah** lists the path name of the current folder.

## Flags

**+folder msgs** Specifies the folder or the messages for which you want to list path names. *msgs* can be several messages, a range of messages, or a single message. You can use the following message references when specifying *msgs*:

<i>num</i>	<b>first</b>	<b>prev</b>	<b>cur</b>
<b>.</b>	<b>next</b>	<b>last</b>	<b>new</b>
<b>all</b>	<i>sequence</i>		

You cannot use **new** in a message range.

If you do not specify a message, **mhpath** lists the path name of the specified folder. The default folder is the current folder.

**-help** Displays help information for the command.

## Profile Entries

**Current-Folder:** Sets your default current folder.  
**Path:** Specifies your *user-mh-directory*.

## Files

`$HOME/.mh-profile` The MH user profile.

## Related Information

The MH command “**folder**” on page 429.

The **mh-mail**, and **mh-profile** files in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

## minidisks

---

## minidisks

---

### Purpose

Adds, deletes, changes, and displays minidisks.

### Syntax

`minidisks` —

OL805307

### Description

The **minidisks** command lets you add, delete, show, or change characteristics of a minidisk. To use the **minidisks** command, you must be a member of the system group or have superuser authority. When a minidisk is added or deleted, an audit record of the type **minidisk - add** or **minidisk - del** is created. When a minidisk is changed, an audit record of the type **stanza - add** and **stanza - del** is created.

The **minidisks** command is menu-driven. For information on how to use it, see *Installing and Customizing the AIX Operating System*.

### Files

/dev	Directory
/tmp	Directory
/etc/ddi	Directory
/etc/master	
/etc/system	
/etc/mdkaf	
/etc/filesystems	
/tmp/CONFIGREPORT	

### Related Information

The following command: “**mkdir**” on page 657.

The discussion of **minidisks** in *Installing and Customizing the AIX Operating System*.

---

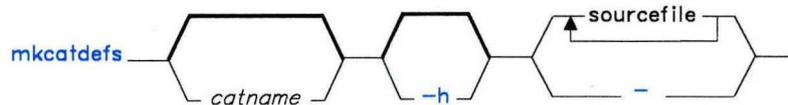
## mkcatdefs

---

### Purpose

Preprocesses a message source file.

### Syntax



OL805486

### Description

The **mkcatdefs** message facility program preprocesses a message source file containing symbolic identifiers<sup>5</sup>, allowing **gencat** to create the **symname** message catalog. The format for **mkcatdefs** is:

```
$ mkcatdefs symname sourcefile
```

The *sourcefile* message source file contains symbolic identifiers.

The **mkcatdefs** program produces the *symname\_msg.h* file containing definition statements equating your symbolic identifiers with set numbers and message ID numbers assigned by **mkcatdefs**. The *symname\_msg.h* file is required in your application program if you use symbolic identifiers.

The **mkcatdefs** program sends message source data, with numbers instead of symbolic identifiers, to standard output. This output is suitable as input to the **gencat** program. You can use the > (redirection symbol) to write the message source to a file, then use the file as input to **gencat**, or use the **runcat** shell script described in “**runcat**” on page 852. You can create a message text source file, using any text editor to enter the messages.

Assign message set numbers and message ID numbers to each message by using the commands described in this section.

---

<sup>5</sup> Symbolic references are not defined by X/Open, but are an AIX extension.

## mkctdefs

---

Use the **\$set** command in a source file to give a group of messages a set number. The format of the **\$set** command is:

```
$set n [comment]
```

The message number is specified by *n*. Instead of a number, you can specify a symbolic identifier, which must contain only letters, digits, or the `_` (underscore character). The maximum length of an identifier is 65 characters. The **mkcatdefs** program assigns a set number to the identifier. The assigned set number is one higher than the preceding set number, or 1 if it is the first **\$set**.

Note that **mkcatdefs** inserts a **\$delset** before the **\$set** in the output message source file.

You can mix numbers and symbolic identifiers.

You can include a comment in the **\$set** command, but it is not required. The following example includes a comment:

```
$set CEM Communication Error Messages
```

Use the **\$delset** command to remove all of the messages belonging to the specified set from a catalog. The format of the **\$delset** command is:

```
$delset n [comment]
```

The message set is specified by *n*. The **\$delset** command must be placed in the proper set number order with respect to any **\$set** commands in the same source file. You can include a comment in the **\$delset** command also.

You can include a comment line anywhere in the source file, except within message text. Indicate comments as shown below:

```
$ [comment]
```

You must leave at least one space after the **\$**.

Enter the message text and symbolic message identifier as follows:

*ID message-text*

*ID* can be either a number or a symbolic identifier and can contain only letters, digits, or the `_` (underscore character). The maximum length of an identifier is 65 characters. The **mkcatdefs** program assigns a message number to the identifier. The assigned number is one higher than the preceding message number, or 1 if it is the first message after the **\$set** command.

Note that **mkcatdefs** inserts a **\$delset** before the **\$set**, which means you cannot add, delete, or replace single messages in the catalog if you are using symbolic message identifiers. You must enter all messages in the set.

You can mix numbers and symbolic identifiers.

You must leave at least one space after the message identifier or number.<sup>6</sup> All text following the first nonblank character is included in the message text, to the end of the line. If the source contains a `$quote` command preceding the message, all text between the two quotation marks is included. Use the `\` (escape character) to continue message text on the following line. The `\` must be the last character on the line, as in the following example:

```
FIVE      Text associated with \  
message FIVE.
```

These two lines define the single-line message:

```
FIVE      Text associated with message FIVE.
```

The `\` can be used to include special characters in the message text. These special characters are defined as follows:

- `\n` Performs a new-line function when the message is displayed.
- `\t` Inserts a horizontal tab character when the message is displayed.
- `\v` Inserts a vertical tab when the message is displayed.
- `\b` Performs a backspace function when the message is displayed.
- `\r` Inserts a carriage-return character when the message is displayed.
- `\f` Inserts a form feed character when the message is displayed.
- `\\` Displays the `\` (backslash) character in the message.
- `\ddd` Displays the single-byte character associated with the octal value represented by the valid octal digits *ddd*. One, two, or three octal digits can be specified; however, you must include leading zeros if the characters following the octal digits are also valid octal digits. For example, the octal value for \$ is 44. To display \$5.00 use `\0445.00`, not `\445.00`, or the 5 will be parsed as part of the octal value.
- `\xddd7` Displays the single-byte or double-byte character associated with the hexadecimal value represented by the four valid hexadecimal digits *ddd*. You can specify one, two, three, or four digits, but you must include leading zeros to avoid parsing errors (see `\ddd`).

---

<sup>6</sup> AIX allows any amount of white space after the message ID number; however X/Open specifies that you leave only one space between the message number and the message text.

<sup>7</sup> This escape sequence is an AIX extension to X/Open.

## mkctdefs

---

You can also include **printf** conversion specifications in messages that are displayed by applications using **printf** or **NLprintf** (see **printf** in *AIX Operating System Technical Reference*). If you display a message from a shell script with **dspmsg**, the message can contain the **%s** or **%n\$s** conversion specifications (see “**dspmsg**” on page 359).

You can use the **\$quote** command in a message source file to define a character for delimiting message text. The format for this command is:

```
$quote [char] [comment]
```

Use the specified character before and after the message text as shown in the following example source file:

```
$quote "      Use a double quotation mark to delimit message text
$set MSFAC          Message Facility - symbolic identifiers
SYM_FORM "Symbolic identifiers can only contain alphanumeric \
characters or the _ (underscore character)\n"
SYM_LEN "Symbolic identifiers cannot be more than 65 \
characters longn \"
5                "You can mix symbolic identifiers and numbers \n"
$quote
MSG_H      Remember to include the "msg_h" file in your program\n
```

In this example, the **\$quote** command sets the quote character to " then disables it before the last message, which contains quotation marks.

When you process this file with **mkcatdefs**, the modified source is written to standard output. Standard output can either be redirected to a file using the **>** (redirection symbol) or piped to **gencat** see “**gencat**” on page 470.

The following source is created:

```
$quote "      Use double quotation marks to delimit message text
```

```
$delset 1
$set 1
```

```
1      "Symbolic identifiers can only contain alphanumeric \
characters or the _ (underscore character)\n"
```

```
2      "Symbolic identifiers cannot be more than 65 \
characters long\n"
```

```
5      "You can mix symbolic identifiers and numbers\n"
```

```
$quote
```

```
6      remember to include the "msg_h" file in your program
```

Note that the assigned message numbers are noncontiguous because the source contained a specific number. The **mkcatdefs** program always assigns the previous number plus 1 to a symbolic identifier.

The **mkcatdefs** program also produces a definition file for inclusion in your program. The name of the file is *symbname*, entered as the first parameter to the **mkcatdefs** command. (If you specify the *-h* flag instead of the *symbname*, no definition file is produced.)

If the symbolic source defined above were in a file called `symb.src`, you could use the **mkcatdefs** command as follows:

```
$ mkcatdefs symb symb.src >symb.msg
```

The generated `symb_msg.h` file would look as follows:

```
#include <limits.h>
#include <nl_types.h>
#define MF_SYMB "symb.cat"
```

```
/* The following was generated from symb.src. */
```

```
/* definitions for set MSFAC */
#define MSFAC 1

#define SYM_FORM 1
#define SYM_LEN 2
#define MSG_H 6
```

## mkctdefs

---

Note that **mkcatdefs** also created a symbol **MF\_SYMB** by adding **MF\_** to the *symbname*, in uppercase letters. The **mkcatdefs** program assumes that the name of the generated catalog should be *symbname.cat*, and generates this symbol for your use with **catopen** or **NLcatopen**.

Since this file includes **limits.h** and **nl\_types.h**, you do not need to include them in your application program. (**nl\_types** defines special data types required by the message facility routines.)

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

## Flags

**-h**           Supresses the generation of a **...msg.h** file.

## Related Information

The following commands: “**dspect**” on page 357, “**dspmsg**” on page 359, “**gencat**” on page 470, and “**runcat**” on page 852.

The **catgets**, **catgetamsg**, **catclose**, **NLcatopen**, **NLcatgets**, and **NLgetamsg** files in *AIX Operating System Technical Reference*.

The discussion of **mkcatdefs** in *AIX Operating System Programming Tools and Interfaces*.

---

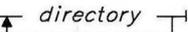
## mkdir

---

### Purpose

Makes a directory.

### Syntax

```
mkdir  directory
```

OL805037

### Description

The **mkdir** command makes a new *directory* in either the local or a remote node. **mkdir** creates the new directories with read, write, and execute permissions enabled for all users. You can change the permissions it sets by default with the **umask** command (see page 1110). **mkdir** also creates by default the standard entries `.` (dot), for the directory itself, and `..` (dot dot), for its parent.

**Note:** To make a new *directory* you must have write permission in the parent directory.

### Related Information

The following commands: “**sh**” on page 913, “**rm**” on page 833, and “**umask**” on page 1110.

The **mkdir** system call in *AIX Operating System Technical Reference*.

# mkfs

---

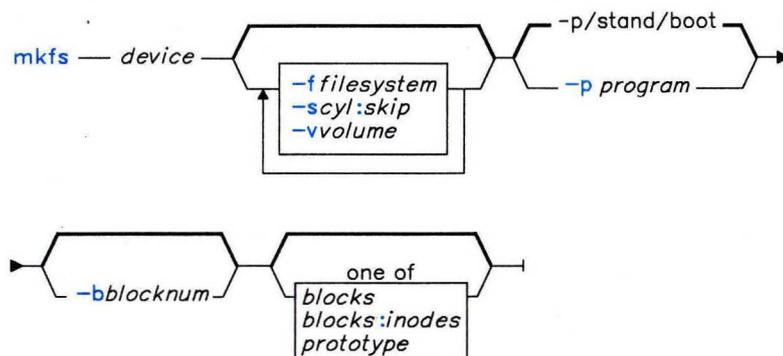
## mkfs

---

### Purpose

Makes a file system.

### Syntax



OL805364

### Description

The **mkfs** command makes new file systems. **mkfs** initializes the volume label and file system label, start-up block, bad-block list, and interleaves the free list in accordance with the flags or with defaults found in the `/etc/filesystems` file.

The **mkfs** command creates the new file system on the *device* specified on the command line. *device* can be a block device name, raw device name, or file system name. If it is a file system name, **mkfs** uses this name as the *file system* and uses the following parameters from the applicable stanza in `/etc/filesystems`:

<b>dev</b>	Device name.
<b>cyl</b>	See the following <code>-s</code> flag.
<b>skip</b>	See the following <code>-s</code> flag.
<b>vol</b>	Volume ID.
<b>bad</b>	List of bad blocks separated by commas.
<b>size</b>	File system size.
<b>boot</b>	Program to be installed in startup block.

## File System Size

You can specify the size of a new file system in the following way:

- On the command line
- In the *prototype* file
- In the */etc/filesystems* entry for the given file system.

If the size is not specified in any of these places, **mkfs** takes it from the **devinfo** structure for the block device associated with the file system being generated. (See the **ioctl** system call and the **devinfo** file in *AIX Operating System Technical Reference*.) The size provided in the **devinfo** structure is the maximum size of the file system in any case. A size specification on the command line overrides any defaults found in the **devinfo** structure or in */etc/filesystems*.

## Prototype Files

To initialize the contents of a new file system in accordance with a prototype, specify the name of a *prototype* file on the command line. The **proto** command can be used to construct prototype files from existing file systems.

The *prototype* file contains tokens separated by spaces or new-line characters. The first token is the name of a file to be copied onto block 0 as the bootstrap program. The second token is a number specifying the size of the created file system. Typically it is the number of blocks on the device, perhaps diminished by space for paging. The next token is the number of i-nodes in the i-list. (**mkfs** rounds this to fill out the appropriate number of blocks.) The next set of tokens contains the specifications for the root file. File specifications consist of tokens giving the mode, the user name, the group name, and the initial contents of the file. The syntax of the contents field depends on the mode.

The mode token for a file is a six-character string. The first character specifies the type of the file. (The characters **-**, **b**, **c**, and **d** specify regular, block special, character special, and directory files, respectively.) The second character must be either **u** or **-**. If **u** is used, the set-user-ID mode is specified; if **-** is used, the set-user-ID mode is not specified. The third character must be either **g** or **-** for specifying the set-group-ID mode. The rest of the mode is a three-digit octal number giving the owner, group, and other read, write, execute permissions (see “**chmod**” on page 160).

Two decimal number tokens come after the mode. They specify the user and group names of the owner of the file.

If the file is a regular file, the next token is a path name from which the contents and size are copied.

If the file is a block or character special file, two decimal number tokens follow, which give the major and minor device numbers.

## mkfs

---

If the file is a directory, **mkfs** makes the entries **.** (dot) and **..** (dot dot) and then recursively reads a list of names and file specifications for the entries in the directory. The scan is ended with the token **\$** (dollar sign).

### Flags

<b>-bblocknum</b>	When present, specifies the number of blocks allocated to file <b>i-node1</b> which is automatically created.
<b>-ffilesystem</b>	Specifies the file system label for the new file system. This can be up to 6 bytes.
<b>-pprogram</b>	Specifies the name of a program to be installed in block 0 of the new file system. The default bootstrap program is <b>/stand/boot</b> .
<b>-scyl:skip</b>	Specifies an interleaving of the free list. (Interleaving the free list can improve the speed of disk I/O.) <i>cyl</i> is the number of blocks per cylinder, and <i>skip</i> is the number of blocks to skip.
<b>-vvolume</b>	Specifies the volume label for the new file system. This can be up to 6 bytes.
<b>blocks[:inodes]</b>	A size specification where <i>blocks</i> is the number of 512-byte blocks in the file system. When <i>inodes</i> is specified, it determines the number of i-nodes on the system. If <i>inodes</i> is not specified, a number suitable for the size of the file system is used. The number of i-nodes is rounded up so that the i-node area occupies an integral number of blocks.

### Examples

1. To create an empty file system on a diskette:

```
mkfs /dev/fd0
```

2. To specify volume and file system names for a new file system:

```
mkfs /dev/fd0 -fWORKFS -vVOL001
```

This creates an empty file system on the diskette, giving it the volume serial VOL001 and file system name WORKFS.

### Related Information

The following command: “**fsck**, **dfsck**” on page 445.

The **ioctl** system call and the **devinfo**, **dir**, **filesystems**, and **fs** files in *AIX Operating System Technical Reference*.

---

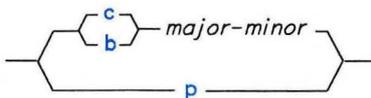
# mknod

---

## Purpose

Creates a special file.

## Syntax

`mknod - device`   `{ c b } major-minor`  
 `{ c b } major-minor`  
 `p`

OL805146

## Description

The **mknod** command makes a directory entry and corresponding i-node for a special file. The first parameter is the name of the entry *device*. Select a name that is descriptive of the device.

The **mknod** command has two forms. In the first case, the second argument is **b** or **c**. The **b** argument indicates that the special file is a block-oriented device (disk, diskette, tape). The **c** argument indicates that it is a character-oriented device (other devices). The last two parameters are numbers specifying the *major* device, which helps the operating system find the device driver code, and the *minor* device, that is, the unit drive, or line number, which may be either decimal or octal.

The assignment of major device numbers is specific to each system. Device numbers are determined by examining the system source file **conf.c**.

**Note:** If you change the contents of **conf.c** to add a device driver, you must rebuild the operating system. See the discussion of device drivers in *AIX Operating System Programming Tools and Interfaces* and in *AIX Operating System Technical Reference*.

The second form of **mknod** is used to create FIFOs (named pipes). The **p** flag after *device* indicates that you are creating a named pipe. See the *AIX Operating System Technical Reference* for an explanation of FIFOs and named pipes.

## **mknod**

---

### **Example**

To create the special file for a new diskette drive:

```
mknod /dev/fd2 b 1 2
```

This creates the special file **/dev/fd2**, which is a block special file with major device number 1 and minor device number 2.

### **Related Information**

The **mknod** file and device driver description in *AIX Operating System Technical Reference*.

The discussion of device drivers in *AIX Operating System Programming Tools and Interfaces*.

---

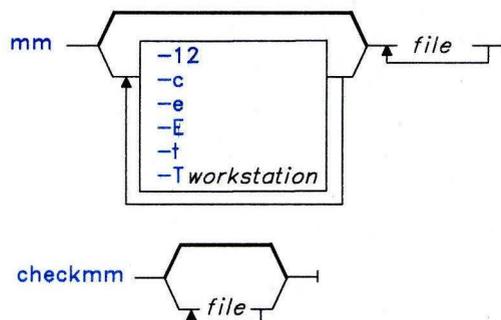
## mm, checkmm

---

### Purpose

Displays or checks documents formatted with Memorandum Macros.

### Syntax



OL805039

### Description

Using the `nroff` command and the Memorandum Macro text-formatting package (MM), the `mm` command writes *files* to standard output. If you specify a - (minus) instead of any *files*, `mm` reads standard input. Do not specify both file names and standard input on the command line.

The `mm` command has flags to specify preprocessing by the `tbl` and/or `eqn` commands and postprocessing by various work station oriented-output filters. It generates the proper pipelines, and the required arguments for `nroff` and MM, depending on the flags selected, creates the required pipelines.

The `checkmm` command is a program for checking the contents of the named *files* for errors in the use of MM and some `eqn` and `neqn` constructions. The program skips all directories, and if you do not specify a file, `checkmm` reads standard input.

### Notes:

1. Use the **-olist** argument of **nroff** to specify ranges of pages to be output. Note, however, that invoking **mm** with one or more of the **-e**, **-t**, and **-** minus arguments together with **nroff -olist** may cause a harmless broken pipe diagnostic if the last page of the document is not specified in *list*.
2. The **mm** command calls **nroff** with the **-h** flag. With this flag, **nroff** assumes that the work station has tabs set every eight character positions.
3. If you use the **-s** flag of **nroff** (to stop between pages of output), use a line feed (rather than **Enter** or a new-line character) to restart the output. The **-s** flag of **nroff** does not work with the **-c** flag of **mm** or if **mm** automatically calls the **col** command.
4. If you provide inaccurate information to **mm** about the kind of work station its output is to be printed on, you will get unsatisfactory results; however, if you are redirecting output to a file, use the **-T37** flag and then use the appropriate work station filter when you actually print the file.

## Flags

Any flags on the command line not listed below are passed to **nroff** or to **MM**, as appropriate. The flags can occur in any order, but they must come before *file*. To obtain a list of **mm** flags, enter the command name with no arguments.

- c** Invokes the **col** command. Note that **col** is invoked automatically by **mm** unless *workstation* (the **-T** flag parameter) is one of the following:
- 300
  - 300s
  - 450
  - 37
  - 4000a
  - 382
  - 4014
  - tek
  - 1620
  - X
- e** Invokes the **neqn** command.
- E** Invokes the **-e** flag of **nroff**.
- t** Invokes the **tbl** command.
- Tworkstation** Uses work station specification *workstation*. For a list of recognized values for *workstation*, enter:
- `help term1`

By default, **mm** uses the value of the shell variable **\$TERM** from the environment as the value of *workstation*. If **\$TERM** is not set **mm** uses **lp**. If several work station types are specified, the last one listed takes effect.

**-12** Uses 12-pitch font. This may be used when **\$TERM** is set to one of **300**, **300s**, **450**, or **1620**. (The pitch switch on the DASI 300 and 300s work stations must be manually set to 12 if this flag is used.)

## Related Information

The following commands: “**col**” on page 179, “**env**” on page 393, “**eqn, neqn, checked**” on page 395, “**greek**” on page 499, “**mmt, checkmm**” on page 666, “**nroff, troff**” on page 709, and “**tbl**” on page 1053.

The **profile** file and the **eqnchar**, **mm**, and **term** miscellaneous facilities in *AIX Operating System Technical Reference*.

The discussion of **mm** in *Text Formatting Guide*.

# mmt

---

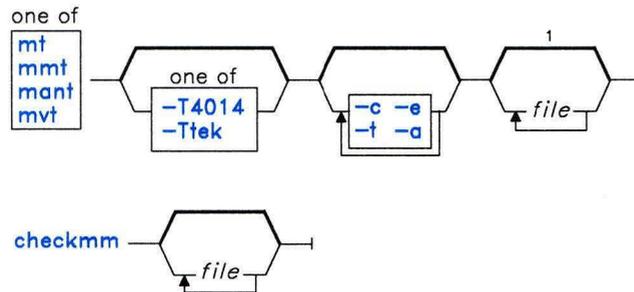
## mmt, checkmm

---

### Purpose

Typesets documents, manual pages, view graphs, and slides.

### Syntax



<sup>1</sup> If no files are given, these commands will display their flags.

OL805092

### Description

These commands are similar to the `mm` command, except they typeset their input via `troff` as opposed to formatting it via `nroff`. The `mvt`, `mt`, and `mant` commands are links to `mmt`. `mmt` uses the MM Macro Package (see `mm` in *AIX Operating System Technical Reference*), `mvt` uses the macro package for view graphs and slides (see `mv` in *AIX Operating System Technical Reference*), `mant` uses the manual page macros, and `mt` does not use a macro package.

These commands have flags to specify preprocessing by `tbl`, `cw`, or `eqn`. `mmt` generates the proper pipelines and the required arguments for `troff` and for the macro package used, depending on the flags selected. These commands read standard input if you specify a - (minus) instead of any file names.

The `checkmm` command can be used to check the input to `mmt`.

If the input contains a **troff** comment line consisting solely of the string `'\ ' x` (single quotation mark, backslash, double quotation mark *x*), where *x* is any combination of the three letters **c**, **e**, and **t** and where there is exactly one blank between the double quotation mark and *x*, then the input will be processed through the appropriate combination of **cw**, **eqn**, and **tbl**, respectively, regardless of the command-line arguments.

**Note:** Use the **-olist** argument of **troff** to specify ranges of pages to be output. Note, however, that calling these commands with one or more of the **-c**, **-e**, **-t**, and **-** arguments together with **troff -olist** may cause a harmless **broken pipe** diagnostic if the last page of the document is not specified in *list*.

## Flags

Flags other than the ones listed below are passed to **troff** or to the macro package, as appropriate. All flags must appear before the *file* names. If you do not provide any arguments, these commands print a list of their flags.

- a**        Invokes the **-a** flag of **troff**.
- c**        Preprocesses the input files with **cw**.
- e**        Preprocesses the input files with **eqn**.
- t**        Preprocesses the input files with **tbl**.
- T4014**
- Ttek**    Directs the output to a Tektronix 4014 work station via the **tc** command.

## Related Information

The following commands: “**env**” on page 393, “**eqn**, **neqn**, **checkeq**” on page 395, “**mm**, **checkmm**” on page 663, “**tbl**” on page 1053, “**tc**” on page 1056, and “**troff**” on page 710.

The **profile** file and the **environ**, **mm**, and **mv** miscellaneous facilities in *AIX Operating System Technical Reference*.

**moo**

---

**moo**

---

## Purpose

Plays a number-guessing game.

## Syntax

`/usr/games/moo` —

OL805231

## Description

The **moo** command picks a random four-digit decimal number with nonrepeating digits. You guess four digits and score a “cow” with a correct digit in an incorrect position and a “bull” with a correct digit in a correct position. The game continues until you guess the number.

To quit the game, press INTERRUPT (**Alt-Pause**) or END OF FILE (**Ctrl-D**).

---

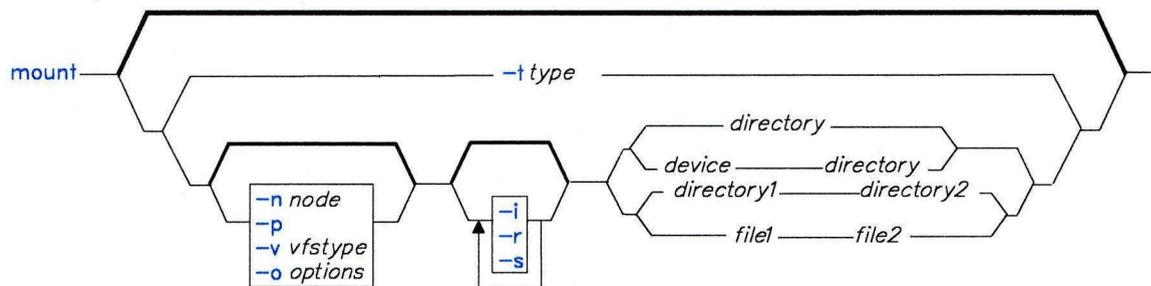
# mount

---

## Purpose

Makes a file system available for use.

## Syntax



OL805467

## Description

The **mount** command instructs the operating system to make a file system available for use. In addition, you can use **mount** to build other file trees made up of directory and file mounts. In the case of file system mounts, the **mount** command mounts the specified *device* on the specified *directory*. After **mount** has finished, *directory* becomes the root of the newly mounted file system.

Any user can issue a **mount** *directory1* *directory2* or **mount** *file1* *file2* command if that user has search or write permission to the directory or file to be mounted over (*directory2* or *file2*). Members of the system group can also do any mount described in the `/etc/filesystems` file (**mount** *directory*). Users operating as superusers can issue any **mount** command.

If you enter the **mount** command without arguments, it writes to standard output the mounted file systems, their locations, and their mount options.

If you specify only a *directory* name, **mount** takes it to be the name of the directory or file on which a file system, directory, or file is usually mounted (as defined in the `/etc/filesystems` file). **mount** looks up the associated device, directory, or file and mounts it. This is the most convenient way of using the **mount** command, as it does not require you to remember what is normally mounted on a directory or file.

## mount

---

The `/etc/filesystems` file should include a *stanza* for each mountable file system, directory, or file. This stanza should specify at least the name of the file system and either the device on which it resides or the directory name. If the stanza includes a *mount attribute*, the `mount` command uses the associated values. It recognizes five values in the mount attribute: **true**, **false**, **removable**, **inherit** and **read-only** (see the `filesystems` file in *AIX Operating System Technical Reference* for a description of these mount attributes.) The command `mount all` causes all file systems with the attribute `mount=true` to be mounted in their normal places. This command is typically used during system initialization.

If you are operating with superuser authority, you can mount a file system arbitrarily by naming both a *device* and a *directory* on the command line. `mount` takes *device* to be the name of the block device special file and *directory* to be the directory on which it should mount the file system.

## Flags

- i** Requests an inherited mount. This option is only valid for Distributed Services mounts. (For information on inherited mounts, see *Managing the AIX Operating System*.)
- n node** Specifies the node that holds the directory to be mounted. If you specify this option without specifying the directory to be mounted, `mount` displays a list of all mounts issued at *node*, if the nodes are connected with Distributed Services.
- o** Sets options for a hard or soft mount in the Network File System environment. The following options can be used:
  - bg** Mount is attempted in background if first attempt fails.
  - fg** Mount is attempted in foreground if first attempt fails.
  - retry = n** Mount is attempted *n* times.
  - rsiz = n** Sets read buffer size to *n* bytes.
  - wsiz = n** Sets write buffer size to *n* bytes.
  - timeo = n** Sets NFS timeout period to *n* tenths of a second.
  - retrans = n** Sets the number of NFS transmissions to *n*.
  - port = n** Sets server IP port number to *n*.
  - soft** Error is returned if server does not respond.
  - hard** Request is retried until server responds.
  - intr** Allows keyboard interrupts on hard mounts.
  - rw** Mounted file is read/write accessible.

**ro** Mounted file is read-only.

The default options and their values (if any) are as follows: **fg**, **retry** = 10000, **rsize** = 8192, **wsize** = 8192, **timeo** = 7, **retrans** = 3, **port** = NFS\_PORT (a system-specified constant), **hard**, and **rw**.

**Note:** Options you enter on the command line should be separated only by a comma, not a comma and a space.

- p** Mounts a file system as a removable file system. While there are open files on it, a removably mounted file system behaves the same as a normally mounted file system. However, when there are no open files (and no process has a current directory on the file system), all of the file system's disk buffers are written to the medium, and the operating system "forgets" the structure of the file system. This allows you to remove and reinsert media such as diskettes without issuing a **mount** or **umount** command each time. Use this flag only for diskette mounts.
- r** Mounts a file system as a read-only file system, regardless of the specification in **/etc/filesystems**.
- s** This flag is for backwards compatibility only.
- t type** Mounts all stanzas in **/etc/filesystems** that contain **type** = *type* and are not mounted. (*type* is a string value, such as *remote*.)
- v vfstype** Mounts the file system using the specified file system type, such as *aix*, *ds*, or *nfs*. If no *vfstype* is specified, the defaults set in the **/etc/vfs** file are used.

## Examples

1. To list the file systems that are mounted:

```
mount
```

nodename	mounted	mounted over	vfs	date	options
-	/dev/hd0	/	aix	Dec 17 08:04	rw
-	/dev/hd6	/vrm	aix	Dec 17 08:05	ro
-	/dev/hd1	/u		Dec 17 08:06	rw
-	/dev/hd2	/usr		Dec 17 08:06	rw
-	/dev/hd3	/tmp		Dec 17 08:06	rw
darlene	/usr	/usr	ds	Dec 17 10:44	rw

For each file system, **mount** lists the node name, the device name, the name under which it is mounted, the access permitted (read only or read/write), and the date and time it was mounted.

## mount

---

2. To mount a diskette:

```
mount /dev/fd0 /diskette0
```

This mounts a diskette (`/dev/fd0`) onto the directory `/diskette0`. A file system must already exist on the diskette, and the directory `/diskette0` must already exist. To access a file on the diskette, use a path name that begins with `/diskette0`. For example, to access `prog.c` use `/diskette0/prog.c`.

**Warning:** Be sure that the current directory is not still on the diskette when you remove it from the drive, or you may lose some of your data.

3. To mount a write-protected diskette:

```
mount -r /dev/fd0 /diskette0
```

This mounts the diskette on `/diskette0` as a read-only file system. This tells the operating system not to update file access times, which would cause errors with a write-protected diskette.

4. To mount a default file system:

```
mount /diskette0
```

This mounts the device that is usually mounted on `/diskette0`, which is determined by information in the file `/etc/filesystems`.

5. To mount all default file systems:

```
mount all
```

This mounts all standard file systems in `/etc/filesystems` marked `mount = true`.

6. To mount a remote directory:

```
mount -n nodeA /u/tom /u/tom
```

This mounts the remote **nodeA** directory `/u/tom` onto the local node directory `/u/tom`. It assumes the default remote **vfs\_type** (which must be defined in `/etc/vfs`).

7. To mount a file or directory from the `/etc/filesystem` file with a specific type:

```
mount -t remote
```

This mounts all files or directories in the `/etc/filesystems` file that have a stanza that contains the attribute `type = remote`.

## Files

<code>/etc/filesystems</code>	Descriptions of mountable file systems.
<code>/etc/vfs</code>	Descriptions of file system types.

## Related Information

The following command: “**umount, unmount**” on page 1112.

The **mount**, **mntctl**, **umount**, and **vmount** system calls and the **vfs** and **filesystems** files in *AIX Operating System Technical Reference*.

The discussion of distributed services, code service, NFS, and the overview of international character support in *Managing the AIX Operating System*.

# mountd

---

## mountd

---

### Purpose

Answers NFS mount requests.

### Syntax

`/usr/etc/rpc.mountd` —|

OL805506

### Description

The **mountd** daemon answers file system mount requests. It reads the file `/etc/exports` to determine which file systems are available to which machines and users in the network.

This daemon also identifies clients that have file systems mounted. Use the **showmount** command to display this information.

The **inetd** daemon invokes the **mountd** daemon.

---

#### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

### Files

`/etc/exports`  
`/etc/inetd.conf`

### Related Information

The following command: “**showmount**” on page 945.

The **inetd** daemon in *IBM RT Interface Program for use with TCP/IP*.

The NFS chapter in *Managing the AIX Operating System*.

---

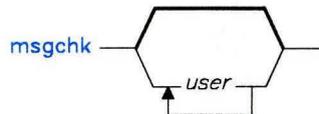
# msgchk

---

## Purpose

Checks for messages.

## Syntax



`msgchk -help`

AJ2FL231

## Description

The **msgchk** command is used to check mail drops for messages waiting to be received. **msgchk** is part of the MH (Message Handling) package and can be used with other MH and AIX commands.

The **msgchk** command checks all mail drops belonging to the specified user IDs and reports which mail drops contain messages that have not been received. **msgchk** also indicates whether it appears you have already seen these messages. If you do not specify a *user* argument, **msgchk** checks the current user's mail drops.

## Flag

**-help** Displays help information for the command.

## Files

<code>\$HOME/.mh-profile</code>	The MH user profile.
<code>/usr/lib/mh/mtstailor</code>	The MH tailor file.
<code>/usr/mail/\$USER</code>	The location of the mail drop.

## Related Information

The MH command “**inc**” on page 518.

The **mh-mail** and **mh-profile** files in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

---

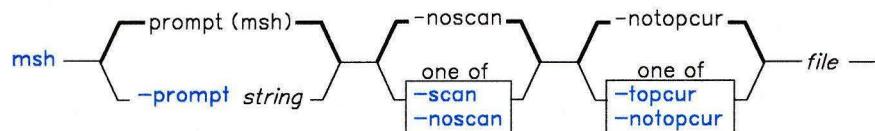
# msh

---

## Purpose

Creates an MH shell.

## Syntax



msh — -help —

AJ2FL232

## Description

The **msh** command is used to work with messages stored in a packed format. **msh** is part of the MH (Message Handling) package and can be used with other MH and AIX commands.

The **msh** command performs a modified subset of MH commands on messages stored in packed format. **msh** prompts you to enter one of the following MH commands, and continues to prompt you for commands until you press **END OF FILE** or enter **quit**:

<b>ali</b>	<b>burst</b>	<b>comp</b>	<b>dist</b>
<b>folder</b>	<b>forw</b>	<b>inc</b>	<b>mark</b>
<b>mhmail</b>	<b>msgchk</b>	<b>next</b>	<b>packf</b>
<b>pick</b>	<b>prev</b>	<b>refile</b>	<b>repl</b>
<b>rmm</b>	<b>scan</b>	<b>send</b>	<b>show</b>
<b>sortm</b>	<b>whatnow</b>	<b>whom</b>	

You can also enter **help** to display a brief overview.

## msh

---

### Flags

<b>-help</b>	Displays help information for the command.
<b>-noscan</b>	Does not scan unseen items.
<b>-notopcur</b>	Makes the current message track the center line of the <b>vmh</b> scan window when <b>msh</b> is invoked from <b>vmh</b> . This flag is the default.
<b>-prompt string</b>	Prompts for <b>msh</b> commands with the specified string.
<b>-scan</b>	Scans unseen items.
<b>-topcur</b>	Makes the current message track the top line of the <b>vmh</b> scan window when <b>msh</b> is invoked from <b>vmh</b> .

### Profile Entries

<b>fileproc:</b>	Specifies the program used to refile messages.
<b>Msg-Protect:</b>	Sets the protection level for your new message files.
<b>Path:</b>	Specifies your <i>user-mh-directory</i> .
<b>showproc:</b>	Specifies the program used to show messages.

### Files

<code>\$HOME/.mh-profile</code>	The MH user profile.
<code>/usr/lib/mh/mtstailor</code>	The MH tailor file.

### Related Information

The following MH commands: “**ali**” on page 48, “**burst**” on page 129, “**comp**” on page 185, “**dist**” on page 336, “**folder**” on page 429, “**forw**” on page 438, “**inc**” on page 518, “**mark**” on page 637, “**mhmail**” on page 646, “**msgchk**” on page 675, “**next**” on page 694, “**packf**” on page 733, “**pick**” on page 748, “**prev**” on page 765, “**refile**” on page 817, “**repl**” on page 821, “**rmm**” on page 841, “**scan**” on page 871, “**send**” on page 893, “**show**” on page 942, “**sortm**” on page 965, “**whatnow**” on page 1215, “**whom**” on page 1222.

The **mh-alias**, **mh-format**, **mh-mail**, and **mh-profile** files in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

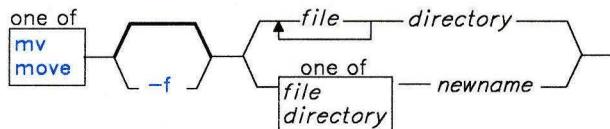
---

**mv**


---

**Purpose**

Moves files.

**Syntax**


OL805010

**Description**

**Warning:** The **mv** command may overwrite many files if you do not ensure that the file path names you are specifying do not already exist.

The **mv** (**move**) command moves files from one directory to another, or it renames a file or directory. If you move a *file* to a new *directory*, it retains the base file name. When you move a file, all links to other files remain intact, except when you move it to a different file system.

You can only rename a *directory* with **mv**; you cannot move it. Both *directory* and *newname* must have the same parent. All files in *directory* are moved to a newly-created directory *newname* under the same file names.

When you use **mv** to rename a *file*, then *newname* can specify either a new file name or a new directory path name. If moving the file would overwrite an existing write-protected file and if standard input is a work station, **mv** displays the permission code of the file to be overwritten and reads one line from standard input. If the line begins with *y*, the move takes place and the file is overwritten. If not, **mv** does nothing with the *file*.

---

**Japanese Language Support Information**

An affirmative response in Japanese Language Support matches one of the elements in the environment variable **YESSTR**.

---

**Note:** If the *file* is on different file system than *directory*, **mv** must copy the *file* to the new file system and delete the original. In this case, the owner name becomes that of the user, and all links to other files are lost.

### Flags

- f Does not prompt before removing a write-protected file.

### Examples

1. To rename a file:

```
mv appendix apndx.a
```

This renames `appendix` to `apndx.a`. If a file named `apndx.a` already exists, its old contents are replaced with those of `appendix`.

2. To rename a directory:

```
mv book manual
```

This renames `book` to `manual`. If a directory named `manual` already exists, then an error message is displayed.

3. To move a file to another directory and give it a new name:

```
mv intro manual/chap1
```

This moves `intro` to `manual/chap1`. The name `intro` is removed from the current directory, and the same file appears as `chap1` in the directory `manual`.

4. To move a file to another directory, keeping the same name:

```
mv chap3 manual
```

This moves `chap3` to `manual/chap3`.

**Note the difference:** Examples 1 and 3 name two files, Example 2 names two existing directories, and Example 4 names a file and a directory.

5. To move several files into another directory:

```
mv chap4 jim/chap5 /u/manual
```

This moves `chap4` to `/u/manual/chap4` and `jim/chap5` to `/u/manual/chap5`.

6. To use `mv` with pattern-matching characters:

```
mv manual/* .
```

This moves all files in the directory `manual` into the current directory (`.`), giving them the same names they had in `manual`. This also empties `manual`. Note that you must type a space between the star and the period.

## Related Information

The following commands: “**chmod**” on page 160, “**ln**” on page 581, and “**rm**” on page 833.

The **rename** system call in *AIX Operating System Technical Reference*.

The discussion of Japanese Language Support in *Japanese Language Support User's Guide*.

## mvdir

---

## mvdir

---

### Purpose

Moves (renames) a directory.

### Syntax

```
mvdir — directory1 — directory2 —
```

OL805137

### Description

The **mvdir** command renames directories within a file system. To use **mvdir**, you must have write permission to *directory1* and *directory2* and to the parent directories of *directory1* and *directory2*. The *directory1* parameter must name an existing directory. If *directory2* does not exist, *directory1* is moved to *directory2*. If *directory2* exists, *directory1* becomes a subdirectory of *directory2*. Neither directory can be a subset of the other.

**Note:** *directory1* and *directory2* may be the names of files. If *directory2* is a file name, it is replaced with *directory1*.

### Example

To rename or move a directory to another location:

```
mvdir appendixes manual
```

If `manual` does not exist, then this renames the directory `appendixes` to `manual`. You can also rename a directory with the **mv** command.

If a directory named `manual` already exists, this moves `appendixes` and its contents to `manual/appendixes`. In other words, `appendixes` becomes a subdirectory of `manual`.

### Related Information

The following commands: “**mkdir**” on page 657 and “**mv**” on page 679.

The **rename** system call in *AIX Operating System Technical Reference*.

---

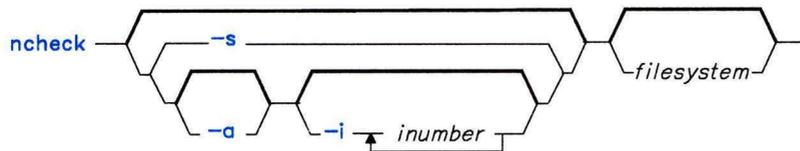
# ncheck

---

## Purpose

Generates path names from i-numbers.

## Syntax



OL805196

## Description

The **ncheck** command without any flags writes to standard output the path name and i-number list for all files in *filesystem*.

If you specify an invalid file system, the ?? in the name stands for the parent of a file system that does not have a parent. Path names beginning with . . . (dot dot dot) indicate a loop.

---

### Japanese Language Support Information

This command has not been modified to support Japanese characters.

---

## Flags

- a** Lists includes the file names . (dot) and .. (dot dot).
- i** *inumber* . . . Lists only the file specified by *inumber*.
- s** Lists only special files and files with set-user-ID mode.

## Examples

1. To list the i-number and path name of each file in the default file systems:  

```
ncheck
```

## ncheck

---

2. To list all the files in a specified file system:

```
ncheck -a /
```

This lists the i-number and path name of each file in the root file system (/), including the . (dot) and .. (dot-dot) entries in each directory (-a).

3. To list the name of a file when you know its i-number:

```
ncheck -i 690 357 280 /diskette0
```

This lists the i-number and path name for every file in the file system **/diskette0** with i-numbers of 690, 357, or 280. If a file has more than one link, all of its path names are listed.

4. To list special and set-user-ID files:

```
ncheck -s /
```

This lists the i-number and path name for every file in the root file system that is a special file (also called a **device file**) or that has set-user-ID mode enabled.

## Related Information

The following commands: “**fsck**, **dfck**” on page 445 and “**sort**” on page 958.

# ndtable

---

## Purpose

Accesses the Distributed Services Node Table.

## Syntax

`ndtable` —

OL805470

## Description

The **ndtable** command lets you build, examine, or change the Distributed Services Network Node Table. Only members of the system group or users operating with superuser authority can use **ndtable** to change the state of the table (see “su” on page 1026). Other users can use **ndtable** to browse through the table.

**Note:** If your system crashes when a user is running **ndtable** to view or update a remote machine’s table, you will get system error message 000-002 the next time you try to access the table. If this happens, remove the contents of the `/etc/profsvcs/NID` directory (where *NID* is the numeric node ID of the remote machine), and retry the **ndtable** command.

## Related Information

“Getting Started With Distributed Services Configuration Menus” in *Managing the AIX Operating System*.

# newform

---

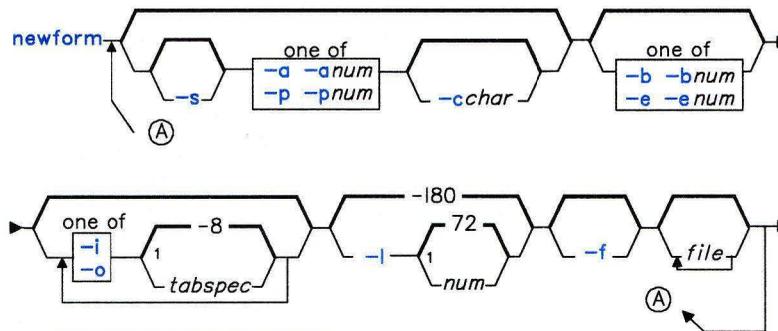
## newform

---

### Purpose

Changes the format of a text file.

### Syntax



<sup>1</sup>Do not put a blank between these items.

OL805197

### Description

The **newform** command takes lines from *file* (standard input by default), and writes the formatted lines to standard output. Lines are reformatted in accordance with the command line flags in effect.

Except for `-s`, command line flags can appear in any order, can be repeated, and can be mixed with the *file* parameter. Command line flags are processed in the order specified. In other words, flag sequences like `-e15 -l60` yield results different from `-l60 -e15`. Flags are applied to all *files* on the command line.

An exit value of 0 indicates normal execution, a 1 indicates an error.

**Notes:**

1. The **newform** command normally only keeps track of physical characters; however, for the **-i** and **-o** flags, **newform** keeps track of backspaces in order to line up tabs in the appropriate logical columns.
  2. The **newform** command does not prompt you if a *tabspec* is to be read from the standard input (by use of **-i--** or **-o --**).
  3. If the **-f** flag is used and the last **-o** flag specified was **-o--** and was preceded by either a **-o--** or a **-i--**, the tab specification format line will be incorrect.
- 

**Japanese Language Support Information**

This command has not been modified to support Japanese characters.

---

**Flags**

- a[num]** Adds *num* characters to the end of the line when the line length is less than the effective line length (see the **-c** and **-p** flags in this section).
- b[num]** Truncates *num* characters from the beginning of the line when the line length is greater than the effective line length (see **-lnum**). The default action truncates the number of characters necessary to obtain the effective line length. If you specify **-b** with no *num*, the default takes effect. This flag can be used to delete the sequence numbers from a COBOL program as follows:
- ```
newform -l1-b7 file-name
```
- The **-ll** must be used to set the effective line length shorter than any existing line in the file so that the **-b** flag is activated.
- c[char]** Changes the prefix/add character to *char*. Default character for *char* is a space.
- e[num]** Same as **-bnum** except that characters are truncated from the end of the line.
- f** Writes the tab specification format line to standard output before any other lines are written. The tab specification format line displayed corresponds to the format specified in the *last* **-o** flag. If no **-o** flag is specified, the line displayed contains the default specification of **-8**.
- i[tabspec]** Replaces all tabs in the input with the number of spaces specified by *tabspec*. *tabspec* recognizes all tab specification forms described in “**tabs**” on page 1041. If you specify a -- (minus minus) for the value of *tabspec*, **newform** assumes that the tab specification can be found in the first line read from standard input (see **fspec** in *AIX Operating System Technical Reference*). The default *tabspec* is **-8**. A *tabspec* of **-0** expects no tabs; if any are found, they are treated as **-1**.

## newform

---

- l[*num*]** Sets the effective line length to *num* characters. If *num* is not entered, **-l** defaults to 72. The default line length without the **-l** flag is 80 characters. Note that tabs and backspaces are considered to be one character (use **-i** to expand tabs to spaces).
- o[*tabspec*]** Replaces spaces in the input with a tab in the output, according to the tab specifications given. The default *tabspec* is -8. A *tabspec* of **-0** means that no spaces are converted to tabs on output.
- p[*num*]** Prefixes *num* characters (see **-cchar**) to the beginning of a line when the line length is less than the effective line length. The default action is to prefix the number of characters that are necessary to obtain the effective line length.
- s** Removes leading characters on each line up to the first tab and places up to eight of the removed characters at the end of the line. If more than eight characters (not counting the first tab) are removed, the eighth character is replaced by an \* (asterisk) and any characters to the right of it are discarded. The first tab is always discarded.

The removed characters are saved internally until all other flags specified are applied to that line. The characters are then added at the end of the processed line.

For example, to convert a file with leading digits, one or more tabs, and text on each line, to a file beginning with the text, all tabs after the first expanded to spaces, padded with spaces out to column 72 (or truncated to column 72), and the leading digits placed starting at column 73, the command would be as follows:

```
newform -s -i -l -a -e file-name
```

The **newform** command displays an error message and stops if this flag is used on a file without a tab on each line.

## Related Information

The following commands: “**tabs**” on page 1041 and “**csplit**” on page 252.

The **fspec** file in *AIX Operating System Technical Reference*.

---

# newgrp

---

## Purpose

Changes your primary group identification.

## Syntax

```
newgrp - [group]
```

OL805198

## Description

The **newgrp** command changes your *primary group* identification to *group*. **newgrp** recognizes only group names, not group ID numbers. Without an argument, it changes your primary group to the one specified in the `/etc/passwd` file.

If the group has a password and you do not or if the group has a password and you are not listed in the `/etc/group` file as a member, then **newgrp** asks you for the group password. (The use of group passwords is not encouraged because, by their very nature, they encourage poor security practices.)

**Note:** Any active user-generated shell will be terminated when **newgrp** is used.

---

### Japanese Language Support Information

This command has not been modified to support Japanese characters.

---

## Flag

- Changes the environment to the login environment of the new group.

## Examples

1. To change the primary group ID of the current shell session to **admin**:  
`newgrp admin`
2. To change the primary group ID back to your original login group:  
`newgrp`

## **newgrp**

---

### **Files**

/etc/group      Group file; contains group IDs.  
/etc/passwd    Password file; contains user IDs.

### **Related Information**

The following commands: “**login**” on page 584 and “**users, adduser**” on page 1129.  
The **group** and **passwd** files in *Installing and Customizing the AIX Operating System*.

---

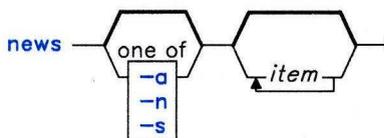
## news

---

### Purpose

Writes system news items to standard output.

### Syntax



OL805199

### Description

The **news** command keeps you informed of news concerning the system. Each news *item* is contained in a separate file in directory **/usr/news**. Anyone having read/write permission to this directory can create a news file.

If you run the **news** command without any flags, it displays every current file in **/usr/news**, showing the most recent first. Or you can specify the *items* you want displayed.

Each file is preceded by an appropriate header. To avoid reporting old news, **news** stores a currency time. **news** considers your currency time to be the modification time of the file named **\$HOME/.news\_time**. Each time you read the news, the modification time of this file changes to that of the reading. Only news item files posted after this time are considered current.

Pressing INTERRUPT (**Alt-Pause**) during the display of a news item stops the display of that item and starts the next. Pressing INTERRUPT (**Alt-Pause**) again ends **news**.

Most users run **news** each time they log in by including the line:

```
news -n
```

in their **\$HOME/.profile** file or in the system's **/etc/profile**.

### Flags

- a** Displays all news items, regardless of the currency time. The currency time does not change.

## news

---

- n Reports the names of current news items without displaying their contents. The currency time does not change.
- s Reports the number of current news items without displaying their names or contents. The currency time does not change.

## Examples

1. To display the items that have been posted since you last read the news:

```
news
```

2. To display all the news items:

```
news -a | pg
```

This displays all the news items a page at a time (`| pg`) whether or not you have read them yet.

3. To list the names of the news items that you have not read yet:

```
news -n
```

Each name is a file in the directory `/usr/news`.

4. To display specific news items:

```
news newusers services
```

This displays news about `newusers` and `services`, which are names listed by `news -n`.

5. To display the number of news items that you have not read yet:

```
news -s
```

6. To post news for everyone to read:

```
cp schedule /usr/news
```

This copies the file `schedule` into the system news directory, `/usr/news`, to create the file `/usr/news/schedule`. To do this you must have write permission for `/usr/news`.

## Files

```
/etc/profile  
/usr/news/*  
$HOME/.news_time
```

## Related Information

The following command: “**pg**” on page 744.

The **profile** file and **environ** special facility in *AIX Operating System Technical Reference*.

## next

---

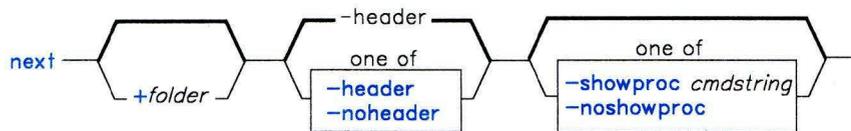
## next

---

### Purpose

Shows the next message.

### Syntax



next — -help —|

AJ2FL159

### Description

The **next** command is used to display the next message in a folder. **next** is equivalent to the **show** command with **next** specified as the message. **next** is part of the MH (Message Handling) package and can be used with other MH and AIX commands.

The **next** command links to the **show** program and passes **show** its flags and attributes.

**Note:** If you link to **next** and call that link something other than **next**, your link will function like the **show** command, rather than like the **next** command.

The **show** command invokes a program to perform the listing. The system default is **/bin/pg**. You can define your own default with the **showproc:** entry in **\$HOME/.mh-profile**. If you set **showproc:** entry to **mhl**, **show** calls an internal **mhl** routine instead of the **mhl** command. You can also specify the program to perform a listing in the *cmdstring* of the **-showproc** flag.

The **show** command passes any flags that it does not recognize to the program performing the listing. Thus, you can specify flags for the listing program, as well as the flags described in this command section.

---

## Flags

|                            |                                                                                                                              |
|----------------------------|------------------------------------------------------------------------------------------------------------------------------|
| <b>+folder</b>             | Specifies the folder that contains the message you want to show.                                                             |
| <b>-header</b>             | Displays a one-line description of the message being shown. The description includes the folder name and the message number. |
| <b>-help</b>               | Displays help information for the command.                                                                                   |
| <b>-noheader</b>           | Does not display a one-line description of each message being shown.                                                         |
| <b>-noshowproc</b>         | Uses <code>/bin/cat</code> to perform the listing.                                                                           |
| <b>-showproc cmdstring</b> | Uses the specified command string to perform the listing.                                                                    |

## Profile Entries

|                        |                                              |
|------------------------|----------------------------------------------|
| <b>Current-Folder:</b> | Sets your default current folder.            |
| <b>Path:</b>           | Specifies your <i>user_mh_directory</i> .    |
| <b>showproc:</b>       | Specifies the program used to show messages. |

## Files

`$HOME/.mh-profile`      The MH user profile.

## Related Information

Other MH commands: “**prev**” on page 765, “**show**” on page 942.

The **mh-format**, **mh-mail**, and **mh-profile** files in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

## nfsd

---

## nfsd

---

### Purpose

Starts the daemons that handle client Network File System requests.

### Syntax

```
nfsd — nservers —|
```

OL805488

### Description

The **nfsd** command starts the daemons that handle client file system requests.

The *nservers* parameter specifies the number of file system request daemons to start. Assign the number based on the load expected on a server. Four is an average load number.

When a file opened by a client is unlinked, a new file is created by the client. The new file is in the form **.nfsxxx**. When the open file is closed, the **.nfsxxx** file is removed.

**Note:** If the client crashes before the file can be closed, the **.nfsxxx** file is not removed.

---

#### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

### Files

**.nfsxxx** Client machine's pointer to an open file that has been unlinked.  
*/etc/rc.nfs*

### Related Information

The following command, “**biod**” on page 114.

The IBM AIX/RT Network File System section in *Managing the AIX Operating System*.

---

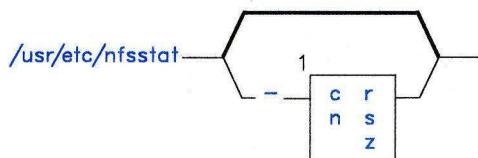
# nfsstat

---

## Purpose

Displays Network File System statistics.

## Syntax



<sup>1</sup> Do not put a blank between these items.

OL805489

OL805308

## Description

The **nfsstat** command displays statistical information about NFS (Network File System) and RPC (Remote Procedure Call).

If you have superuser authority, you can also use **nfsstat** to reinitialize the NFS and RPC statistical information.

The default for **nfsstat** is **nfsstat -csnr**. If you use it without flags, information for NFS and RPC is displayed and reinitialization does not occur.

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

## Flags

Use the following flags for displaying and reinitializing the statistics for specific areas:

- c Displays NFS and RPC client information.
- cn Displays NFS client information only.
- cr Displays RPC client information only.

## nfsstat

---

- n Displays NFS client and server information.
- r Displays RPC client and server information.
- s Displays NFS and RPC server information.
- sn Displays NFS server information only.
- sr Displays RPC server information only.
- z Reinitializes statistics to zero. Only users with superuser authority can reinitialize statistics.

After displaying certain sets of the statistics, you can reinitialize them by using the flag that identifies the area followed by the **-z** flag. For example, you would use the flag combination **-cnz** to reinitialize NFS client statistics.

## Files

/unix System name list  
/dev/kmem Kernel memory

## Related Information

The following command: “**rstatd**” on page 847.

---

## nice

---

### Purpose

Runs a command at a different priority.

### Syntax

```
nice -[number] cmdstring
```

<sup>1</sup>Maximum increment is 19.

OL805200

### Description

The **nice** command lets you run the specified *command* at a lower priority. The value of *number* can range from 1 to 19, with 19 being the lowest priority. The default value of *number* is 10.

If you have superuser authority, you can run commands at a higher priority by specifying *number* as a negative number, such as `--10`.

### Examples

1. To run a command at low priority:

```
nice cc -c *.c
```

This runs the command `cc -c *.c` at low priority. Note that this does not run the command in the background. Your work station is not available for doing other things.

2. To run a low priority command in the background:

```
nice cc -c *.c &
```

This runs the command `cc -c *.c` at low priority in the background. Your work station is free so that you can run other commands while `cc` is running. See page 914 for details about starting background processes with `&`.

## nice

---

3. To specify a very low priority:

```
nice -15 cc -c *.c &
```

This runs **cc** in the background at a priority that is even lower than the default priority set by **nice**.

4. To specify a very high priority:

```
nice --10 wall <<end  
System shutdown in 2 minutes!  
end
```

This runs **wall** at a higher priority than all user processes. Doing this slows down everything else running on the system. If you do not have superuser authority when you run this command, then the **wall** command runs at the normal priority.

The <<end and end define a “Here Document,” which uses the text entered before the end line as standard input for the command. For more details, see “Inline Input Documents” on page 928.

## Related Information

The following commands: “**cs**h” on page 225 and “**no**hup” on page 707.

**Note:** The **cs**h command contains a built-in subcommand named **nice**. The command and subcommand do not necessarily work the same way. For information on the subcommand, see the **cs**h command.

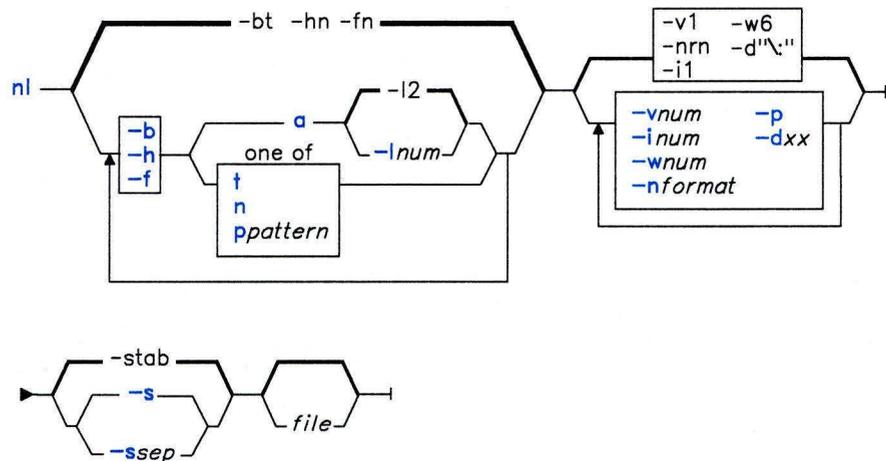
The **nice** system call in *AIX Operating System Technical Reference*.

## nl

## Purpose

Numbers lines in a file.

## Syntax



OL805386

## Description

The `nl` command reads *files* (standard input by default), numbers the lines in the input, and writes the numbered lines to standard output. In the output, `nl` numbers the lines on the left according to the flags you specify on the command line.

The input test must be written in logical pages. Each logical page has a header, a body, and a footer section (you can have empty sections). Unless you use the `-p` flag, `nl` resets the line numbers at the start of each logical page. You can set line numbering flags independently for the header, body, and footer sections (for example, no numbering of header and footer lines while numbering text lines only in the body).

Signal the start of logical page sections with lines in *file* that contain nothing but the following delimiter characters:

| <i>Line contents</i> | <i>Start of</i> |
|----------------------|-----------------|
| \: \:                | Header          |
| \: :                 | Body            |
| \:                   | Footer          |

You can name only one file on the command line. You can list the flags and the file name in any order.

## Flags

All the parameters are set by default. Use the following flags to change these default settings. Except for the **-s** flag, enter a flag without a parameter to see its default value.

- btype** Chooses which body section lines to number. The recognized *types* are:
- a** Numbers all lines.
  - t** Does not number blank lines (default).
  - n** Does not number any lines.
  - ppattern** Numbers only those lines containing the specified *pattern*.
- dxx** Uses *xx* as the delimiters for the start of a logical page section. The default characters are \: (backslash followed by a colon). You may specify two ASCII characters, two 1-byte extended characters, or one extended character. If you enter only one 1-byte character after **-d**, the second character remains the default (colon). If you want to use a backslash as a delimiter, enter two backslashes (\\).
- ftype** Chooses which logical page footer lines to number. The *types* recognized are the same as in **-btype**. The default *type* is **n** (no lines numbered).
- htype** Chooses which logical page header lines to number. The *types* recognized are the same as in **-btype**. The default *type* is **n** (no lines numbered).
- inum** Increments logical page line numbers by *num*. The default value of *num* is 1.
- lnum** Uses *num* as the number of blank lines to count as one. For example, **-l3** will only number the third adjacent blank. The default value of *num* is 2. This flag can only be used in documents where the **-ba** flag is used.
- nformat** Uses *format* as the line numbering format. Recognized formats are:
- ln** Left justified, leading zeros suppressed.
  - rn** Right justified, leading zeros suppressed (default).
  - rz** Right justified, leading zeros kept.
- p** Does not restart numbering at logical page delimiters.

- 
- `-s[sep]` Separates the text from its line number by the *sep* character. The default value of *sep* is a tab character. If you enter `-s` without a parameter, there is no separation between the line number and its text.
  - `-vnum` Sets the initial logical page line number to *num*, (1 by default).
  - `-wnum` Uses *num* as the number of characters in the line number. The default value of *num* is 6.

## Examples

1. To number only the nonblank lines:

```
nl chap1
```

This displays a numbered listing of `chap1`, numbering only the nonblank lines in the body sections. If `chap1` contains no `\:|\+ ;`, `\:|\+ :`, or `\:` delimiters, then the entire file is considered the body.

2. To number all lines:

```
nl -ba chap1
```

This numbers all the lines in the body sections, including blank lines. This form of the `nl` command is adequate for most uses.

3. To specify a different line number format:

```
nl -i10 -nrz -s:: -v10 -w4 chap1
```

This numbers the lines of `chap1` starting with ten (`-v10`) and counting by tens (`-i10`). It displays four digits for each number (`-w4`), including leading zeros (`-nrz`). The line numbers are separated from the text by two colons (`-s::`).

For example, if `chap1` contains the text:

```
A not-so-important
note to remember:

You can't kill time
without injuring eternity.
```

then the numbered listing is:

```
0010::A not-so-important
0020::note to remember:

0030::You can't kill time
0040::without injuring eternity.
```

Note that the blank line was not numbered. To do this, use the `-ba` flag as shown in Example 2.

**nl**

---

## **Related Information**

The following command: “**pr**” on page 761.

“Overview of International Character Support” in *Managing the AIX Operating System*.

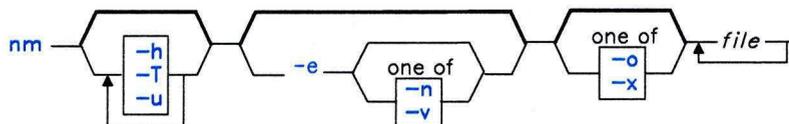
---

**nm**


---

**Purpose**

Displays the symbol table of an object file.

**Syntax**

OL805202

**Description**

The **nm** command writes the symbol table of each specified object *file* to standard output. *file* can be a single relocatable or absolute common object file or an archive library of relocatable or absolute common object files. **nm** displays the following information for each symbol:

|                |                                                                                                                                                                                                                                                                                                         |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Name</b>    | The name of the symbol.                                                                                                                                                                                                                                                                                 |
| <b>Value</b>   | Its value expressed as an offset or an address depending on its storage class.                                                                                                                                                                                                                          |
| <b>Class</b>   | Its storage class.                                                                                                                                                                                                                                                                                      |
| <b>Type</b>    | Its type and derived type. If the symbol refers to a structure or a union, the structure or union tag follows the type declaration. If the symbol is an array, the array dimensions follow the type. Note that you must have compiled the object file with <b>cc -g</b> for this information to appear. |
| <b>Size</b>    | Its size in bytes, if available. Note that you must have compiled the object file with <b>cc -g</b> for this information to appear.                                                                                                                                                                     |
| <b>Line</b>    | The source line number at which it is defined, if available. Note that you must have compiled the object file with <b>cc -g</b> for this information to appear.                                                                                                                                         |
| <b>Section</b> | For static and external storage classes, the object file section containing the symbol.                                                                                                                                                                                                                 |

## nm

---

### Flag

- e Displays only static and external symbols.
- h Does not display output header data.
- n Sorts external symbols by name before displaying them. Use this flag only in conjunction with the -e flag.
- o Displays a symbol's value and size as an octal rather than a decimal number.
- T Truncates every name that would otherwise overflow its column, making the last character displayed in the name an asterisk. By default, **nm** displays the entire name of the symbols listed, and a name that is longer than the width of the column set aside for it causes every column after the name to be misaligned.
- u Displays only undefined symbols.
- v Sorts external symbols by value before displaying them. Use this flag only in conjunction with the -e flag.
- x Displays a symbol's value and size as a hexadecimal rather than a decimal number.

### Files

a.out Default input file.

### Related Information

The following commands: “**ar**” on page 55, “**as**” on page 61, “**backup**” on page 88, “**cc**” on page 140, “**ld**” on page 557, “**size**” on page 949, and “**strip**” on page 1017.

The **a.out** and **ar** files in *AIX Operating System Technical Reference*.

# nohup

---

## Purpose

Runs a command without hangups and quits.

## Syntax

```
nohup — command —
```

OL805203

## Description

The **nohup** command runs *command*, ignoring all hangups and **QUIT** signals. You can use this command to run programs in the background after you log off of the system. To run a **nohup** command in the background, add an **&** to the end of the command.

If **nohup** output is redirected to a terminal or is not redirected at all, the output goes to the file **nohup.out**. If **nohup.out** is not writable in the current directory, the output is redirected to **\$HOME/nohup.out**.

The syntax of this command ignores quits and hangups for only one *command*. If you want to apply **nohup** to a pipeline or list of commands, you can put the pipeline or list in a shell script file. Then you can run **sh** as the *command* using the format: `nohup sh file`. You can also assign the shell file execute permission and run it as the command in the form: `nohup file`.

## Examples

1. To leave a command running after you log off:

```
nohup find / -print &
```

Shortly after you enter this, the following is displayed:

```
670  
$ Sending output to nohup.out
```

## nohup

---

The number will probably be different when you use this command. It is the ID of the background process started by & (ampersand). (See page 914 about starting background processes with &.) The \$ (dollar sign) is your shell prompt. Sending output . . . is a message from **nohup** telling you that it is storing the output from the `.find` command in the file `nohup.out`. You can log off after you see these messages, even if the `find` command has not finished yet.

2. To do the same, but redirect the standard output to a different file:

```
nohup find / -print >filenames &
```

This runs the `find` command and stores its output in a file named `filenames`. Now only the process ID and your prompt are displayed:

```
677  
$
```

Wait for a second or two before logging off, because the **nohup** command takes a moment to start the *command* you specified. If you log off too quickly, your *command* may not run at all. Once your *command* has started, logging off will not affect it.

3. To run more than one command, use a shell procedure. For example, if you write the shell procedure:

```
neqn math1 ] nroff > fmath1
```

and name it `nnfmath1`, you can run **nohup** for all of the commands in `nnfmath1` with the command:

```
nohup sh nmath1
```

If you assign `nnfmath1` execute permission, you can obtain the same results by issuing the command:

```
nohup nnmath1
```

To run this command in the background, enter the command:

```
nohup nnmath1 &
```

## Related Information

The following commands: “**cs**h” on page 225, “**n**ice” on page 699, and “**s**h” on page 913.

**Note:** The **cs**h command contains a built-in subcommand named **nohup**. The command and subcommand do not necessarily work the same way. For information on the subcommand, see the **cs**h command.

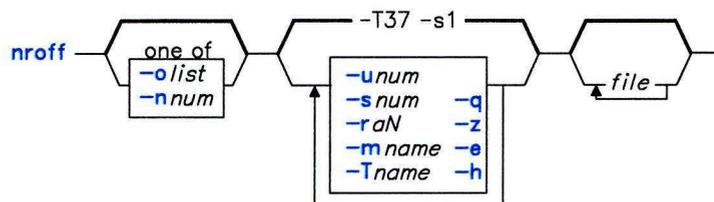
The **signal** system call in *AIX Operating System Technical Reference*.

# nroff, troff

## Purpose

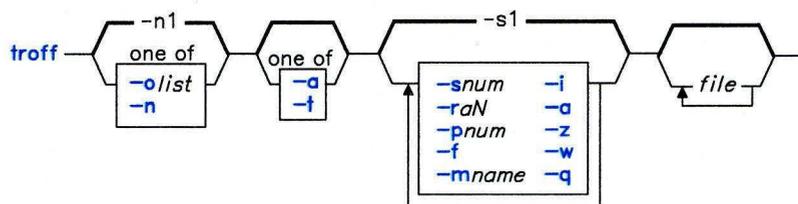
Formats text for printing devices.

## Syntax



OL805204

**troff** `-b`



OL805368

## Description

A complete list of **nroff** and **troff** requests, escape sequences and number registers begins on page 712. See *Text Formatting Guide* for a complete list of the naming conventions for the non-ASCII special characters and for information on writing text suitable for processing by **troff** or **nroff**.

## nroff, troff

The **nroff** command reads *files* (standard input by default), formats the text in its input for printing, and writes to standard output. **nroff** formats text for line printers and other **printing devices**, excluding phototypesetters. An input file name of - (minus) indicates standard input.

## troff

The **troff** command formats text in the input *files* (or standard input by default) for a phototypesetter, and writes its output to standard output. It is similar to the **nroff** command. An input file name of - (minus) indicates standard input.

## nroff and troff Flags:

- i** Reads standard input after the input files.
- mname** Adds `/usr/lib/tmac/tmac.name` to the beginning of the list of input file names.
- num** Numbers the first printed page *num*. Do not use this flag with **-olist**.
- olist** Prints only pages with page numbers appearing in *list* which consist of a comma-separated list of page numbers and ranges. A range of *A-B* means print pages *A* through *B*; an initial *-A* means print from the beginning to page *A*; and a final *A-* means print from page *A* to the end.  
**Note:** When this flag is used in a pipeline (for example, with **cw**, **eqn**, or **tbl**) it may cause a broken pipe diagnostic if the last page in the document is not specified in *list*.
- q** Invokes the simultaneous input/output mode of the **.rd** request. **nroff** echoes the **.rd** prompt, but does not echo your input. When you enter two consecutive new-line characters, normal output is resumed.
- raN** Sets register *a* to *N*. *a* must have a one-character name. This is useful for automatic numbering of sections, paragraphs, lines, and so forth.
- snum** Stops every *num* pages (the default is 1). **nroff** or **troff** will halt every *num* pages to allow paper loading or changing and will resume upon receipt of a line-feed or new-line character. This flag does not work in pipelines. When **nroff** halts between pages, an ASCII BEL character is sent to the printing device.
- z** Suppresses the formatted output. Prints only messages generated by **.tm** (work station message) requests.

**nroff Flags:**

- e** Produces equally spaced words in adjusted lines, using the full resolution of the printing device.
- h** Uses tab characters during horizontal spacing. Tab settings are assumed to be every eight spaces.
- Tname** Prepares the output for the specified printing device. Known *names* are:
  - 37** TELETYPE Model 37 work station (default)
  - tn300** GE TermiNet 300 or any work station without half-line capability
  - 300s** DASI 300s
  - 300** DASI 300
  - 450** DASI 450
  - lp** Any ASCII line printer
  - 382** DCT-382
  - 4000A** Trendata 4000A
  - 832** Anderson Jacobson 832
  - X** any EBCDIC printer
  - 2631** Hewlett Packard 2631 line printer. Use a *name* of **2631-c** to get compressed print. Use **2631-e** to get expanded print.
- u[num]** Sets the number of character overstrikes for boldface to *num* or to zero if *num* is not specified.

**troff Flags:**

- a** Sends a printable ASCII approximation of the output to standard output.
- b** Reports whether the phototypesetter is busy or available. No text processing is done.
- f** Does not feed out paper and stop the phototypesetter at the end of the run.
- pnum** Prints all characters in the point size specified by *num*. Smaller point sizes may reduce the printing time.
- t** Directs output without modification to standard output instead of the phototypesetter.
- w** Waits until phototypesetter is available if it is currently busy.

## nroff

---

### nroff and troff Requests

| Request Form           | Function -- Font and Character Size Control                                         |
|------------------------|-------------------------------------------------------------------------------------|
| <code>.ps ± N</code>   | Change point size by $N$ points. Also, for <b>troff</b> only, <code> s ± N</code> . |
| <code>.ss N</code>     | Space-character size set to $N/36$ em ( <b>troff</b> only).                         |
| <code>.cs F N M</code> | Constant character space (width) mode (font $F$ ) ( <b>troff</b> only).             |
| <code>.bd F N</code>   | Embolden font $F$ by $N$ units ( <b>troff</b> only).                                |
| <code>.bd S F N</code> | Embolden Special Font when current font is $F$ ( <b>troff</b> only).                |
| <code>.ft F</code>     | Change to font $F$ .                                                                |
| <code>.fp N F</code>   | Mount font $F$ on position $N$ (1-4).                                               |

| Request Form         | Function -- Page Control                       |
|----------------------|------------------------------------------------|
| <code>.pl ± N</code> | Change page length by $N$ .                    |
| <code>.bp ± N</code> | Eject current page; next page number is $N$ .  |
| <code>.pn N</code>   | Next page number is $N$ .                      |
| <code>.po ± N</code> | Page offset = $N$ .                            |
| <code>.ne N</code>   | Need $N$ vertical space.                       |
| <code>.mk R</code>   | Mark current vertical place in register $R$ .  |
| <code>.rt ± N</code> | Return (upward only) to marked vertical place. |

| Request Form         | Function -- Text Filling, Adjusting, and Centering |
|----------------------|----------------------------------------------------|
| <code>.br</code>     | Break.                                             |
| <code>.fi</code>     | Fill subsequent output lines.                      |
| <code>.nf</code>     | No filling or adjusting of output lines.           |
| <code>.ad [c]</code> | Adjust output lines with mode $c$ .                |
| <code>.na</code>     | Do not adjust output lines.                        |
| <code>.ce N</code>   | Center the following $N$ lines.                    |

| Request Form              | Function -- Vertical Spacing                                     |
|---------------------------|------------------------------------------------------------------|
| <code>.vs <i>N</i></code> | Set vertical base-line spacing to <i>N</i> .                     |
| <code>.ls <i>N</i></code> | Output <i>N</i> -1 base-line spaces after each text output line. |
| <code>.sp <i>N</i></code> | Space vertical distance <i>N</i> in either direction.            |
| <code>.sv <i>N</i></code> | Save vertical distance <i>N</i> .                                |
| <code>.os</code>          | Output saved vertical space.                                     |
| <code>.ns</code>          | Turn no-space mode on.                                           |
| <code>.rs</code>          | Restore spacing; turn no-space mode off.                         |

| Request Form                        | Function -- Line Length and Indenting            |
|-------------------------------------|--------------------------------------------------|
| <code>.li <math>\pm N</math></code> | Change line length by <i>N</i> .                 |
| <code>.in <math>\pm N</math></code> | Change indenting by <i>N</i> .                   |
| <code>.ti <math>\pm N</math></code> | Change the indent on the next line by <i>N</i> . |

| Request Form                      | Function -- Macros, Strings, Diversion, and Position Traps          |
|-----------------------------------|---------------------------------------------------------------------|
| <code>.de <i>xx yy</i></code>     | Define or redefine macro <i>xx</i> ; end at call of <i>yy</i> .     |
| <code>.am <i>xx yy</i></code>     | Append to a macro.                                                  |
| <code>.ds <i>xx string</i></code> | Define a string <i>xx</i> containing <i>string</i> .                |
| <code>.as <i>xx string</i></code> | Append <i>string</i> to string <i>xx</i> .                          |
| <code>.rm <i>xx</i></code>        | Remove request, macro, or string named <i>xx</i> .                  |
| <code>.rn <i>xx yy</i></code>     | Rename request, macro, or string <i>xx</i> to <i>yy</i> .           |
| <code>.di <i>xx</i></code>        | Divert output to macro <i>xx</i> .                                  |
| <code>.da <i>xx</i></code>        | Divert and append to <i>xx</i> .                                    |
| <code>.wh <i>N xx</i></code>      | Set location trap; negative is with respect to the end of the page. |
| <code>.ch <i>xx N</i></code>      | Change trap location.                                               |
| <code>.dt <i>N xx</i></code>      | Set a diversion trap.                                               |
| <code>.it <i>N xx</i></code>      | Set an input line trap.                                             |
| <code>.em <i>xx</i></code>        | End macro is <i>xx</i> .                                            |

## nroff

---

| Request Form                            | Function -- Number Registers                                 |
|-----------------------------------------|--------------------------------------------------------------|
| <code>.nr <math>R \pm N M</math></code> | Define and set number register $R$ ; auto-increment by $M$ . |
| <code>.af <math>R c</math></code>       | Assign format to register $R$ ( $c=1, i, I, a, A$ ).         |
| <code>.rr <math>R</math></code>         | Remove register $R$ .                                        |

| Request Form                           | Function -- Tabs, Leaders, and Fields                            |
|----------------------------------------|------------------------------------------------------------------|
| <code>.ta <math>Nt . . .</math></code> | Tab settings; left type, unless $t=R$ (right) or $C$ (centered). |
| <code>.tc <math>c</math></code>        | Tab repetition character.                                        |
| <code>.lc <math>c</math></code>        | Leader repetition character.                                     |
| <code>.fc <math>a b</math></code>      | Set field delimiter $a$ and pad character $b$ .                  |

| Request Form                             | Function -- Input/Output Conventions and Character Translations                     |
|------------------------------------------|-------------------------------------------------------------------------------------|
| <code>.ec <math>c</math></code>          | Set escape character.                                                               |
| <code>.eo</code>                         | Turn off escape character mechanism.                                                |
| <code>.lg <math>N</math></code>          | Ligature on if $N > 0$ .                                                            |
| <code>.ul <math>N</math></code>          | Underline in <b>nroff</b> or italicize in <b>troff</b> the next $N$ input lines.    |
| <code>.cu <math>N</math></code>          | Continuous underline in <b>nroff</b> . Acts like <code>.ul</code> in <b>troff</b> . |
| <code>.uf <math>F</math></code>          | Underline font set to $F$ (to be switched to by <code>.ul</code> ).                 |
| <code>.cc <math>c</math></code>          | Set control character to $c$ .                                                      |
| <code>.c2 <math>c</math></code>          | Set no-break control character to $c$ .                                             |
| <code>.tr <math>abcd . . .</math></code> | Translates $a$ to $b$ , and so on, on output.                                       |

| Request Form                           | Function -- Hyphenation               |
|----------------------------------------|---------------------------------------|
| <code>.nh</code>                       | No hyphenation.                       |
| <code>.hy <math>H</math></code>        | Hyphenate; $N = \text{mode}$ .        |
| <code>.hc <math>c</math></code>        | Hyphenation indicator character $c$ . |
| <code>.wc<br/><i>word . . .</i></code> | Exception words.                      |

| Request Form                                       | Function -- Three Part Titles |
|----------------------------------------------------|-------------------------------|
| .tl ' <i>left</i> ' <i>center</i> ' <i>right</i> ' | Three part title.             |
| .pc <i>c</i>                                       | Page number character.        |
| .lt $\pm N$                                        | Length of title.              |

| Request Form      | Function -- Output Line Numbering      |
|-------------------|----------------------------------------|
| .nm $\pm N M S I$ | Number mode on or off, set parameters. |
| .nm <i>N</i>      | Do not number next <i>N</i> lines.     |

| Request Form                                              | Function -- Conditional Acceptance of Input                                                                         |
|-----------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------|
| .if <i>c.anything</i>                                     | If condition <i>c</i> is true, accept <i>anything</i> as input. For multiple lines, use <code>\{anything\}</code> . |
| .if ! <i>c anything</i>                                   | If condition <i>c</i> is false, accept <i>anything</i> as input.                                                    |
| .if <i>N anything</i>                                     | If expression $N > 0$ , accept <i>anything</i> as input.                                                            |
| .if ! <i>N anything</i>                                   | If expression $N \leq 0$ , accept <i>anything</i> as input.                                                         |
| .if ' <i>string1</i> ' <i>string2</i> ' <i>anything</i>   | If <i>string1</i> is identical to <i>string2</i> , accept <i>anything</i> as input.                                 |
| .if !' <i>string1</i> ' <i>string2</i> .' <i>anything</i> | If <i>string1</i> is not identical to <i>string2</i> , accept <i>anything</i> as input.                             |
| .ie <i>c anything</i>                                     | If part of <b>if-else</b> ; can take all forms of <b>if</b> above.                                                  |
| .el <i>anything</i>                                       | Else part of <b>if-else</b> .                                                                                       |

| Request Form | Function -- Environment Switching |
|--------------|-----------------------------------|
| .ev <i>N</i> | Environment switched (push down). |

| Request Form      | Function -- Insertions from Standard Input |
|-------------------|--------------------------------------------|
| .rd <i>prompt</i> | Read insertion.                            |
| .ex               | Exit from <b>nroff</b> or <b>troff</b> .   |

## nroff

---

| Request Form             | Function -- Input/Output File Switching             |
|--------------------------|-----------------------------------------------------|
| <code>.so file</code>    | Switch source file (push down).                     |
| <code>.nx file</code>    | Next file.                                          |
| <code>.pi program</code> | Pipe output to <i>program</i> ( <b>nroff</b> only). |

| Request Form             | Function -- Miscellaneous                                                                                          |
|--------------------------|--------------------------------------------------------------------------------------------------------------------|
| <code>.mc c N</code>     | Set margin character <i>c</i> and separation <i>N</i> .                                                            |
| <code>.tm string</code>  | Print <i>string</i> on standard error output.                                                                      |
| <code>.ig yy</code>      | Ignore until call of <i>yy</i> .                                                                                   |
| <code>.pm t</code>       | Print macro names and sizes; if <i>t</i> is present, print only the total of sizes.                                |
| <code>.fl</code>         | Flush output buffer.                                                                                               |
| <code>.ab [text]</code>  | Prints <i>text</i> on standard error output and stops output. User abort is printed if no <i>text</i> is included. |
| <code>! cmd parms</code> | Runs the AIX command <i>cmd</i> and interpolates at that point. The standard input for <i>cmd</i> is closed.       |

## Escape Sequences for Characters, Indicators, and Functions

| Escape Sequence       | Meaning                                                      |
|-----------------------|--------------------------------------------------------------|
| <code>\\</code>       | Prevents or delays interpretation of <code>\</code> .        |
| <code>\e</code>       | Printable version of the current escape character.           |
| <code>\'</code>       | Acute accent; equivalent to <code>\(aa</code> .              |
| <code>\`</code>       | Grave accent; equivalent to <code>\(ga</code> .              |
| <code>\-</code>       | Minus sign in the current font.                              |
| <code>\.</code>       | Dot.                                                         |
| <code>\(space)</code> | Unpaddable space-size character.                             |
| <code>\0</code>       | Digit width space.                                           |
| <code>\ </code>       | 1/6 em narrow space character (zero width in <b>nroff</b> ). |

| Escape Sequence | Meaning                                                                                                           |
|-----------------|-------------------------------------------------------------------------------------------------------------------|
| \^              | 1/12 em half-narrow space character (zero width in <b>nroff</b> ).                                                |
| \&              | Non-printing, zero-width character.                                                                               |
| \!              | Transparent line indicator.                                                                                       |
| \\$N            | Interpolate argument $1 \leq N \leq 9$ .                                                                          |
| \%              | Default optional hyphenation character.                                                                           |
| \(xx            | Character named <i>xx</i> .                                                                                       |
| \ *x, \ *(xx    | Interpolate string <i>x</i> or <i>xx</i> .                                                                        |
| \a              | Non-interpreted leader character.                                                                                 |
| \b'abc . . . '  | Bracket building function.                                                                                        |
| \c              | Interrupt text processing (continue word across input line break).                                                |
| \d              | Forward (down) 1/2 em vertical motion (1/2 line in <b>nroff</b> ).                                                |
| \fx, \f(xx, \fN | Change to font <i>N</i> named <i>x</i> or <i>xx</i> , or font position <i>N</i> .                                 |
| \gx, \g(xx      | Return the format of register <i>x</i> or <i>xx</i> . Return nothing if the register has not yet been referenced. |
| \h'N'           | Local horizontal motion; move right <i>N</i> (negative left).                                                     |
| \jx, \jxx       | Mark in register <i>x</i> or <i>xx</i> the current horizontal position on the output line.                        |
| \kx             | Mark horizontal input place in register <i>x</i> .                                                                |
| \l'N[c]'        | Horizontal line drawing function.                                                                                 |
| \L'N[c]'        | Vertical line drawing function.                                                                                   |
| \nx, \(xx       | Interpolate number register <i>x</i> or <i>xx</i> .                                                               |
| \o'abc . . . '  | Overstrike characters <i>a</i> , <i>b</i> , <i>c</i> , . . . .                                                    |
| \p              | Break and spread output line.                                                                                     |
| \r              | Reverse 1 em vertical motion (reverse line in <b>nroff</b> ).                                                     |
| \sN, \s±N       | Point-size change function.                                                                                       |
| \t              | Non-interpreted horizontal tab.                                                                                   |
| \u              | Reverse (up) 1/2 em vertical motion (1/2 line in <b>nroff</b> ).                                                  |
| \v'N'           | Local vertical motion ; move down <i>N</i> (negative up).                                                         |
| \w'string'      | Interpolate width of <i>string</i> .                                                                              |

## nroff

---

| Escape Sequence          | Meaning                                                      |
|--------------------------|--------------------------------------------------------------|
| <code>\x'N'</code>       | Extra line-space function (negative before; positive after). |
| <code>\zc</code>         | Print <i>c</i> with zero width without spacing.              |
| <code>\{</code>          | Begin conditional input.                                     |
| <code>\}</code>          | End conditional input.                                       |
| <code>\(new line)</code> | Concealed new line.                                          |
| <code>\X</code>          | <i>X</i> , any character not listed above.                   |

## Predefined General Number Registers

| Register Name   | Description                                                     |
|-----------------|-----------------------------------------------------------------|
| <code>%</code>  | Current page number.                                            |
| <code>ct</code> | Character width type (set by width function).                   |
| <code>dl</code> | Maximum width of last completed diversion.                      |
| <code>dn</code> | Height (vertical size) of last completed diversion.             |
| <code>dw</code> | Current day of the week (1=Sunday . . . 7=Saturday).            |
| <code>dy</code> | Current day of the month (1-31).                                |
| <code>hp</code> | Current horizontal place on the input line.                     |
| <code>ln</code> | Output line number.                                             |
| <code>mo</code> | Current month (1-12).                                           |
| <code>nl</code> | Vertical position of last printed text baseline.                |
| <code>sb</code> | Depth of string below base line (generated by width function).  |
| <code>st</code> | Height of string above base line (generated by width function). |
| <code>yr</code> | Last two digits of current year.                                |

---

## Predefined Read-Only Number Registers

| Register Name | Meaning                                                                                                                               |
|---------------|---------------------------------------------------------------------------------------------------------------------------------------|
| \$            | Number of arguments available at the current macro level.                                                                             |
| .A            | Set to 1 in <b>troff</b> if the <b>-a</b> flag is used; always 1 in <b>nroff</b> .                                                    |
| .F            | The name of the current input file.                                                                                                   |
| .H            | Available horizontal resolution in basic units.                                                                                       |
| .L            | Contains the current line spacing parameter.                                                                                          |
| .P            | Contains the value 1 if the current page is being printed, 0 otherwise.                                                               |
| .R            | The number of columns available.                                                                                                      |
| .T            | Set to 1 in <b>nroff</b> , if the <b>-T</b> flag is used; always 0 in <b>troff</b> .                                                  |
| .V            | Available vertical resolution in basic units.                                                                                         |
| .a            | Post-line extra line-space most recently utilized using <code>\s'N'</code> .                                                          |
| .b            | Emboldening factor of the current font.                                                                                               |
| .c            | Number of lines read from current input file, including <b>.so</b> files.                                                             |
| .d            | Current vertical place in current diversion; equal to <b>nl</b> if no diversion.                                                      |
| .f            | Current font as physical quadrant.                                                                                                    |
| .h            | Text base-line high-water mark on current page or diversion.                                                                          |
| .i            | Current indent.                                                                                                                       |
| .j            | Current adjustment mode and type.                                                                                                     |
| .k            | Contains the horizontal size of the text portion of the current, partially-collected output line, if any, in the current environment. |
| .l            | Current line length.                                                                                                                  |
| .n            | Length of text portion on previous output line.                                                                                       |
| .o            | Current page offset.                                                                                                                  |
| .p            | Current page length.                                                                                                                  |
| .s            | Current point size.                                                                                                                   |
| .t            | Distance to the next trap.                                                                                                            |
| .u            | Equal to 1 in fill mode; equal to 0 in no-fill mode.                                                                                  |

## nroff

---

| Register Name | Meaning                              |
|---------------|--------------------------------------|
| .v            | Current vertical line spacing.       |
| .w            | Width of previous character.         |
| .x            | Reserved version-dependent register. |
| .y            | Reserved version-dependent register. |
| .z            | Name of current diversion.           |

## Files

|                      |                                                |
|----------------------|------------------------------------------------|
| /usr/lib/suftab      | Suffix hyphenation tables.                     |
| /tmp/ta\$#           | Temporary file.                                |
| /usr/lib/tmac/tmac.* | Standard macro files.                          |
| /usr/lib/macros/*    | Standard macro files.                          |
| /usr/lib/font/*      | Font width tables for <b>troff</b> .           |
| /usr/lib/term/*      | Work station driving tables for <b>nroff</b> . |

## Related Information

The following commands: “**col**” on page 179, “**cw, checkcw**” on page 275, “**eqn, neqn, checkeq**” on page 395, “**mm, checkmm**” on page 663, “**mmt, checkmm**” on page 666, “**greek**” on page 499, “**tbl**” on page 1053, and “**tc**” on page 1056.

The **mm** miscellaneous facility in *AIX Operating System Technical Reference*.

The discussion of **nroff** and **troff** in *Text Formatting Guide*.

# number

---

## Purpose

Displays the written form of a number.

## Syntax

`/usr/games/number` —

OL805229

## Description

The **number** game displays the written form of a number that it reads from standard input. The largest number it can translate accurately contains 66 digits.

The **number** game does not prompt you for a number. Once loaded, it simply waits for input. To exit the program, press INTERRUPT (**Alt-Pause**) or END OF FILE (**Ctrl-D**).

## Example

To display the written form of several numbers:

You: `/usr/games/number`  
829

System: eight hundred twenty nine.

...  
You: `12345678`

System: twelve million.  
three hundred forty five thousand.  
six hundred seventy eight.

...  
You: **Ctrl-D**

**number**

---

---

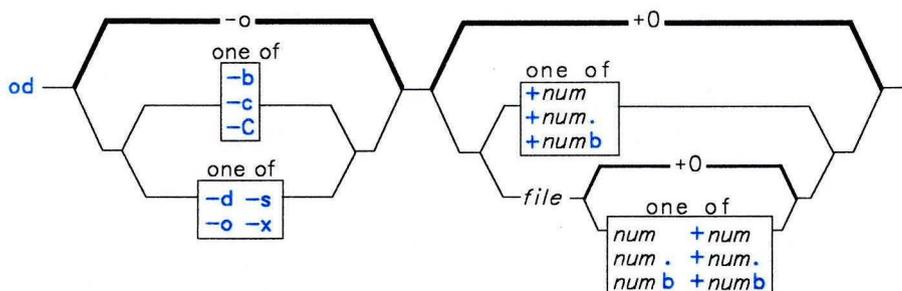
# od

---

## Purpose

Writes the contents of storage to the standard output.

## Syntax



OL805205

<sup>1</sup> Do not put a blank between these items.

OL805308

## Description

The **od** command reads *file* (standard input by default), and it writes to standard output the information stored in *file* using the format specified by the first flag. If you do not specify the first flag, **-o** is the default.

When **od** reads standard input, *num* must be preceded by a + (plus sign).

## Flags

- b** Displays bytes as octal values.
- c** Displays bytes as ASCII characters. The following nongraphic characters appear as C escape sequences:
  - \0** Null
  - \b** Backspace
  - \f** Form feed
  - \n** New-line character

## od

---

`\r` Return  
`\t` Tab  
`\s1`  
`\s2`  
`\s3`  
`\s4` Extended character shifts. (When Japanese Language Support is installed on your system, extended character shifts are not supported.)

Others appear as three-digit octal numbers.

- C Displays any extended characters as standard printable ASCII characters, using the appropriate character escape string.
- 

### Japanese Language Support Information

- C Displays any SJIS characters in hexadecimal form.
- 

- d Displays 16-bit words as unsigned decimal values.
- o Displays 16-bit words as octal values.
- s Displays 16-bit words as signed decimal values.
- x Displays 16-bit words as hexadecimal values.

The *num* parameter specifies the point in the file where the storage output starts. The *num* parameter is interpreted as octal bytes. If a . (dot) is added to *num*, it is interpreted in decimal. If **b** is added to *num*, it is interpreted in blocks of 512 bytes.

The storage output continues until the end of the file.

## Examples

1. To display a file in octal a page at a time:

```
od a.out | pg
```

This displays **a.out** in octal (base 8) word format a page at a time.

2. To translate a file into several formats at once:

```
od -cx a.out >a.xcd
```

This writes **a.out** in hexadecimal (base 16) format (-x) into the file **a.xcd**, giving also the ASCII character equivalent, if any, of each byte (-c).

3. To start in the middle of a file:

```
od -bcx a.out +100.
```

This displays `a.out` in octal-byte, character, and hexadecimal formats, starting from the 100th byte. The `.` (dot) after the offset makes it a decimal number. Without the dot, the dump would start from the 64th (100 octal) byte.

## Related Information

The following commands: “**sdb**” on page 875 and “**pg**” on page 744.

“Overview of International Character Support” in *Managing the AIX Operating System*.

The discussion of Japanese Language Support in *Japanese Language Support User's Guide*.

**on**

---

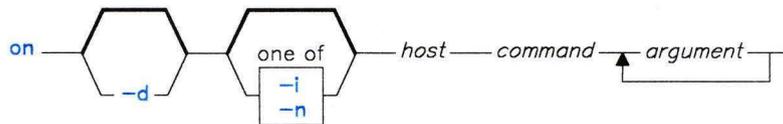
**on**

---

## Purpose

Executes a command remotely when NFS is installed on your system.

## Syntax



OL805490

## Description

The **on** program executes commands on another system. All environment variables are passed. The current working directory is preserved if the working file is mounted on the host or exported to it. Relative path names work only if they are within the current file system. Because commands are issued at one machine and executed on another, absolute path names can cause problems.

The **on** command's standard input passes as standard input to the remote program, unless you use the **-n** flag. The **on** command's files receive standard output and standard error from the remote command.

**Note:** The **on** command cannot be used to execute commands requiring superuser authority.

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

## Flags

- d** Displays status messages as work progresses.
- i** Uses remote echoing and special character processing. This flag should be used when remotely executing interactive programs, such as **vi**. Terminal modes are active when this flag is used.

**-n** Does not pass standard input to the remote standard input. This causes the remote program to get an `end-of-file` message when it reads standard input. The **on -n** program is used to run commands in the background with job control.

## File

`/etc/exports`

## Related Information

The following command, “**rexid**” on page 832.

## open

---

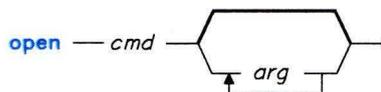
## open

---

### Purpose

Opens a virtual terminal.

### Syntax



OL805337

### Description

The **open** command opens a virtual terminal and runs the specified *file* on that terminal. If *file* does not reside in one of the directories specified in the shell **PATH** variable, you must give a full path name. Any arguments that follow *file* on the command line are passed to *file*. To move from one virtual terminal to another, press Next Window (**Alt-Action**). To close a virtual terminal, press **END OF FILE (Ctrl-D)** or end the application that is running on it.

#### Notes:

1. Be sure to use **Alt-Action** to check for open virtual terminals and **Ctrl-D** to close them before you log off.
2. You can run the **actman** command before opening any virtual terminals to help you keep track of virtual terminals so you can close all of them before you log off.
3. If you are in a trusted shell and issue the command **opensh**, then the new virtual terminal is opened, but not activated. You must use **Alt-Action** to activate the new virtual terminal.

### Usability Services Commands

The following additional commands are available to you from within the Usability Services Activity Manager (**/usr/bin/actmgr**):

- |                 |                                                  |
|-----------------|--------------------------------------------------|
| <b>hide</b>     | Removes an activity window from the window ring. |
| <b>activate</b> | Activates an activity window.                    |
| <b>cancel</b>   | Cancels an activity window.                      |

For details about using these commands, see *Usability Services Reference* or *Usability Services Guide*.

## Example

To run another shell on a new virtual terminal:

```
open sh
```

To move back and forth between this new virtual terminal and any others that you have opened, press Next Window (**Alt-Action**). To close this terminal and log off the new shell, press END OF FILE (**Ctrl-D**).

## Related Information

The following command: “**actman**” on page 32.

“Using Display Station Features” in *Using the AIX Operating System*.

# pack

---

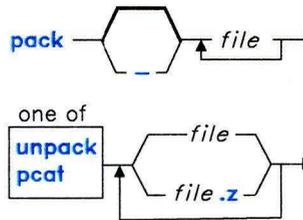
## pack

---

### Purpose

Compresses files.

### Syntax



OL805061

### Description

#### **pack**

The **pack** command stores the specified *file* in a compressed form. The input file is replaced by a packed file with a name derived from the original file name (*file.z*), with the same access modes, access and modification dates, and owner as the original file. Directories cannot be compressed.

The input file name can contain no more than 12 bytes to allow space for the added *z* extension.

If **pack** cannot create a smaller file, it stops processing and reports that it is unable to save space. (A failure to save space, generally happens with small files or files with uniform character distribution.) The amount of space saved depends on the size of the input file and the character frequency distribution. Because a decoding tree forms the first part of each *.z* file, you will generally not be able to save space with files smaller than three blocks. Typically, text files are reduced 25 to 40 percent.

The exit value of the **pack** command is the number of files that it could not pack. Packing is not done under any one of the following conditions:

- The file is already packed.
- The input file name has more than 12 bytes.
- The file has links.
- The file is a directory.
- The file cannot be opened.

- No storage blocks are saved by packing.
- A file called *file.z* already exists.
- The *.z* file cannot be created.
- An I/O error occurs during processing.

**Note:** Both **pcat** and **unpack** operate only on files ending in *.z*. As a result, when you specify a file name that does not end in *.z*, **pcat** and **unpack** add the suffix and search the directory for a file name with that suffix.

### **pcat**

The **pcat** command reads the specified *files*, unpacks them, and writes them to standard output. (Japanese Language Support does not support the **pcat** command.)

### **unpack**

The **unpack** is the reverse of the **pack** command. It reads the input *files*, expands them, and writes them to their original file name—that is, the name without the *.z* suffix.

The exit value of **pcat** is the number of files it was unable to unpack. A file cannot be unpacked if any one of the following occurs:

- The file name (exclusive of the *.z*) has more than 12 bytes.
- The file cannot be opened.
- The file is not a packed file.

The **unpack** command expands files created by **pack**. For each file specified, **unpack** searches for a file called *file.z*. If this file is a packed file, **unpack** replaces it by its expanded version. The **unpack** command names the new file name by removing the *.z* suffix from *file*. The new file has the same access modes, access and modification dates, and owner as the original packed file.

The exit value is the number of files the **unpack** command was unable to unpack. A file cannot be unpacked if any one of the following occurs:

- The file cannot be opened.
- The file is not a packed file.
- A file with the unpacked file name already exists.
- The unpacked file cannot be created.

**Note:** The **unpack** command writes a warning to standard output if the file it is unpacking has links. The new unpacked file has a different i-node than the packed file from which it was created. However, any other files linked to the packed file's original i-node still exist and are still packed.

# pack

---

## Flag

- Displays statistics about the input *files*. The statistics are calculated from a Huffman minimum redundancy code tree built on a byte-by-byte basis. Repeating - (minus) on the command line toggles this function.

## Examples

1. To compress files:

```
pack chap1 chap2
```

This compresses chap1 and chap2, replacing them with files named chap1.z and chap2.z. **pack** displays the percent decrease in size for each file.

2. To display statistics about the amount of compression done:

```
pack - chap1 - chap2
```

This compresses chap1 and chap2 and displays statistics about chap1, but not about chap2. The first - (minus) turns on the statistic display, and the second turns it off.

3. To display compressed files:

```
pcat chap1.z chap2 | pg
```

This displays the compressed files chap1.z and chap2.z on the screen in expanded form, a page at a time (`| pg`). Note that **pcat** added the `.z` to the end of chap2, even though we did not enter it.

4. To use a compressed file without expanding the copy stored on disk:

```
pcat chap1.z | grep 'Greece'
```

This pipes the contents of chap1.z in its expanded form to the **grep** command. See page 914 for a discussion of piping.

5. To expand compressed files:

```
unpack chap1.z chap2
```

This expands the compressed files chap1.z and chap2.z, replacing them with files named chap1 and chap2. Note that you can give **unpack** file names either with or without the `.z` suffix.

## Related Information

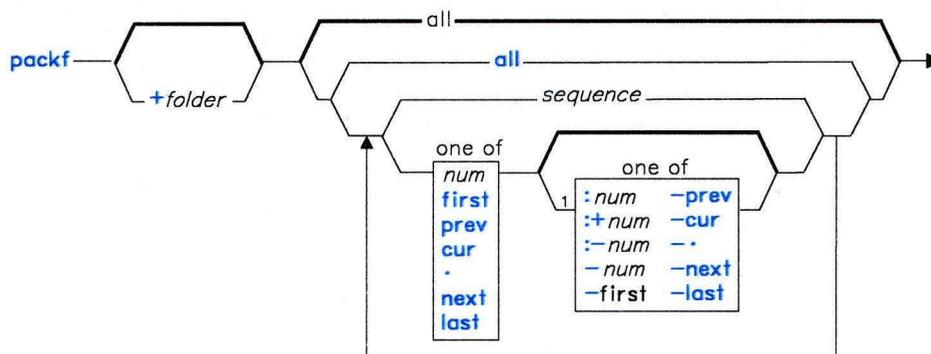
The following command: “**cat**” on page 137.

# packf

## Purpose

Compresses the contents of a folder into a file.

## Syntax



AJ2FL212



`packf -help`

AJ2FL213

<sup>1</sup> Do not put a blank between these items.

OL805308

## Description

The **packf** command is used to move messages from a folder and compress those messages into a single file. You can unpack packed messages by using the **inc** command. **packf** is part of the MH (Message Handling) package and can be used with other MH and AIX commands.

## packf

---

The **packf** command takes the specified messages from the specified folder and appends them to the specified file. If the file does not exist, **packf** displays a prompt that asks if you want to create the file. If you want to create the file, **packf** creates the file and places the messages in the file. **packf** separates each message with four **Ctrl-A** characters and a **New line** character.

### Flags

**-file file** Places the messages at the end of the specified file. If *file* does not exist, **packf** asks you if you want to create the file. If you want to create the file, **packf** creates *file*. The default file is **./msgbox**.

**+folder msgs** Specifies the messages that you want to pack. *msgs* can be several messages, a range of messages, or a single message. You can use the following message references when specifying *msgs*:

|             |              |                 |
|-------------|--------------|-----------------|
| <i>num</i>  | <b>first</b> | <b>prev</b>     |
| <b>cur</b>  | <b>.</b>     | <b>next</b>     |
| <b>last</b> | <b>all</b>   | <i>sequence</i> |

The default is all of the messages in the current folder. If several messages are specified, the first message packed becomes the current message.

**-help** Displays help information for the command.

### Profile Entries

**Current-Folder:** Sets your default current folder.  
**Msg-Protect:** Sets the protection level for your new message files.  
**Path:** Specifies your *user\_mh\_directory*.

### Files

`$HOME/.mh_profile` The MH user profile.

### Related Information

The MH command “**inc**” on page 518.

The **mh-profile** file in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

# passwd

---

## Purpose

Changes login password.

## Syntax

```
passwd — user —
```

OL805206

## Description

The **passwd** command establishes or changes the password for your login user name. When you enter this command, you get a prompt for the old password if one exists, and two successive prompts for the new password. You must enter the same password twice for it to take effect. Passwords can be up to eight ASCII characters long. Only the password owner or the superuser can change a password. To change a password, the owner must know the old password. Passwords are subject to restrictions established by the person who administers the system.

**Note:** The password files must be on the same node.

## Files

|                       |                                                 |
|-----------------------|-------------------------------------------------|
| /etc/passwd           | Password file; contains user IDs.               |
| /etc/opasswd          | Previous version of the password file.          |
| /etc/security/passwd  | Password field; contains encrypted passwords.   |
| /etc/security/opasswd | Previous version of the security password file. |
| /etc/security/config  | Defines password restrictions.                  |

## Related Information

The following commands: “**login**” on page 584 and “**users, adduser**” on page 1129.

The **getpwent** subroutine and the **passwd** file in *AIX Operating System Technical Reference*.

The discussion of password security in *Managing the AIX Operating System*.

The discussion of Japanese Language Support in *Japanese Language Support User's Guide*.

## paste

---

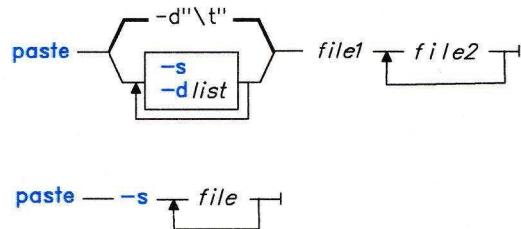
## paste

---

### Purpose

Merges the lines of several files or subsequent lines in one file.

### Syntax



OL805207

OL805366

### Description

The **paste** command reads input *files* (standard input if you specify a - as a file name), concatenates the corresponding lines of the given input files, and writes the resulting lines to standard output. Output lines are restricted to 511 characters.

Without a flag, or with the **-d** flag, **paste** treats each file as a column and joins them horizontally with a tab character by default (parallel merging). You can think of **paste** as the counterpart of the **cat** command (see page 137), which concatenates files vertically, that is, one file after another.

With the **-s** flag, **paste** combines subsequent lines of an input file (serial merging). These lines are joined with the tab character by default.

**Note:** The action of **pr -t -m** is similar to that of **paste**, but creates extra blanks, tabs and lines for a nice page layout.

---

#### Japanese Language Support Information

This command has not been modified to support Japanese characters.

---

## Flags

- dlist** Changes the delimiter that separates corresponding lines in the output with one or more characters in *list* (the default is a tab). If more than one character is in *list*, then they are repeated in order until the end of the output. In parallel merging, the lines from the last file always end with a new-line character, instead of one from *list*.

The following special characters can also be used in *list*:

```
\n      New-line character
\t      Tab
\\      Backslash
\0      Empty string (not a null character).
c       An extended character.
```

You must quote characters that have special meaning to the shell.

- s** Merges subsequent lines from the first file horizontally. With this flag, **paste** works through one entire file before starting on the next. When it finishes merging the lines in one file, it forces a new line and then merges the lines in the next input file, continuing in the same way through the remaining input files, one at a time. A tab separates the lines unless you use the **-d** flag. Regardless of the *list*, the last character of the file is forced to be a new-line character.

## Examples

1. To paste several columns of data together:

```
paste names places dates > npd
```

This creates a file named `npd` that contains the data from `names` in one column, `places` in another, and `dates` in a third. If `names`, `places`, and `dates` look like:

| names  | places     | dates      |
|--------|------------|------------|
| rachel | New York   | February 5 |
| jerry  | Austin     | March 13   |
| mark   | Chicago    | June 21    |
| marsha | Boca Raton | July 16    |
| scott  | Seattle    | November 4 |

then `npd` contains:

## paste

---

```
rachel New York      February 5
jerry  Austin March 13
mark   Chicago June 21
marsha Boca Raton   July 16
scott  Seattle November 4
```

A tab character separates the name, place, and date on each line. As in this example, the columns do not always line up because the tab stops are set at every eighth column.

2. To separate the columns with a character other than a tab:

```
paste -d"!@" names places dates > npd
```

This alternates ! and @ as the column separators. If names, places, and dates are the same as in Example 1, then npd contains:

```
rachel!New York@February 5
jerry!Austin@March 13
mark!Chicago@June 21
marsha!Boca Raton@July 16
scott!Seattle@November 4
```

3. To display the standard input in multiple columns:

```
ls | paste - - - -
```

This lists the current directory in four columns. Each - tells **paste** to create a column containing data read from the standard input. The first line is put in the first column, the second line in the second column, . . . , the fifth line in the first column, and so on.

This is equivalent to:

```
ls | paste -d"\t\t\t\n" -s -
```

which fills the columns across the page with subsequent lines from the standard input. The -d"\t\t\t\n" defines the character to insert after each column: a tab character (\t) after the first three columns, and a new-line character (\n) after the fourth. Without the -d flag, **paste -s -** would display all of the input as one line with a tab between each column.

## Related Information

The following commands: “**grep**” on page 501, “**cut**” on page 269, and “**pr**” on page 761. “Overview of International Character Support” in *Managing the AIX Operating System*.

---

## pcnfs

---

### Purpose

Serves PC-NFS client requests.

### Syntax

`/etc/rpc.pcnfsd` 

OL805499

### Description

The **pcnfsd** daemon handles PC-NFS client requests for services, such as authentication and print spooling, on remote machines. PC-NFS is a program that allows machines running the Personal Computer Disk Operating System (PC-DOS) to be networked with machines running NFS. See *Managing the AIX Operating System* for information on PC-NFS.

The **pcnfsd** daemon starts when the **inetd** server starts if the appropriate entry is placed in the **inetd.conf** file. Print spooling requests are then sent to the default print default print spooling directory, which is usually requests is If you plan to use a directory other than the default print spooling directory (**/usr/tmp**), you cannot start **pcnfsd** from **inetd.conf**. Instead, you must specify the directory (using the **-2** flag) and start **pcnfsd** from the command line.

---

#### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

### Flags

**-s spooldir** Specifies the name of the directory designated to receive print spooling requests for PC-NFS clients. When this flag is used, the **pcnfsd** command must be issued from the command line. If the **spooldir** name is not specified, the default print spooling directory is used.

## **pcnfsd**

---

### **File**

`/etc/inetd.conf` TCP/IP configuration file.

### **Related Information**

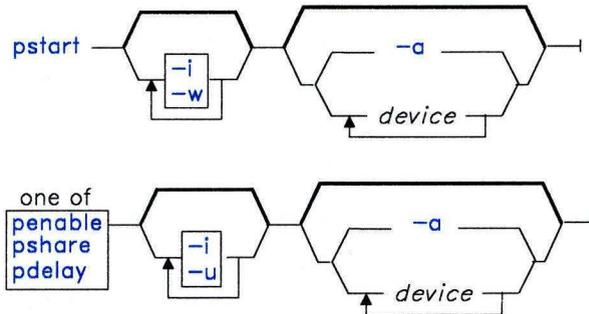
The section on configuring **pcnfsd** in *Managing the AIX Operating System*.

# pdisable, phold

## Purpose

Disables or reports the availability of login ports.

## Syntax



OL805208

## Description

The **pdisable** and **phold** commands each disable a set of login ports. No one can log in on a disabled port. The system disables a port by updating an entry in the `/etc/portstatus` file and then sending a signal to **init**. When **init** receives the signal and reads the updated status entry, it takes the appropriate action.

Use the *device* parameter to specify the ports to be enabled. Permitted values for *device* include:

- A full device name, such as `/dev/tty1`.
- A simple device name, such as `tty1`.
- A general class of devices in the form `attribute=value`, which is equivalent to naming each port with a stanza in `/etc/ports` that includes the specified attribute).

If you do not specify a *device* to disable, each command report the names of currently disabled ports in its set.

## pdisable

---

### pdisable

The **pdisable** command disables the specified port. Even if a user is logged in at that port, subsequent log in is not permitted. To disable the port, the system ends **logger**. If you do not specify any arguments, **pdisable** reports the names of all disabled ports.

### phold

The **phold** command allows logged-in users to continue, but does not allow any more users to log in. If you do not specify any arguments, **phold** reports the names of all ports on hold.

## Flags

- a With **pdisable**, disables all ports that are currently enabled in the **/etc/portstatus** file. With **phold**, holds all ports that are currently enabled in the **/etc/portstatus** file.
- i Reinitializes an existing **/etc/portstatus** file instead of updating the existing one. You typically use this flag in the **/etc/rc** command file to re-establish default port enabling before starting up the system with multiple users.
- w Makes the command return immediately rather than wait for **init** to confirm the changes in port status.

## Examples

1. To display the names of all ports currently disabled:  
`pdisable`
2. To disable all ports that are enabled in **/etc/portstatus**, even if users are logged in:  
`pdisable -a`
3. To disable the work station attached to the **/dev/tty8** port:  
`pdisable tty8`
4. To disable the work station attached to the **/dev/tty2** and make the command return immediately rather than wait for **init** to confirm port status:  
`pdisable -w /dev/tty2`
5. To list the ports that are currently on hold:  
`phold`
6. To put all 9600 baud ports on hold:  
`phold speed=9600`

## Files

|                 |                                                                         |
|-----------------|-------------------------------------------------------------------------|
| /etc/locks      | Contains lock files for <b>pshare</b> and <b>pdelay</b> .               |
| /etc/ports      | Contains descriptions of known normal, shared, and delayed login ports. |
| /etc/portstatus | Contains current status of each known login port.                       |

## Related Information

The following commands: “**init**” on page 521 and “**pstart, penable, pshare, pdelay**” on page 791.

The **ports** and **portstatus** files in *AIX Operating System Technical Reference*.

## pg

---

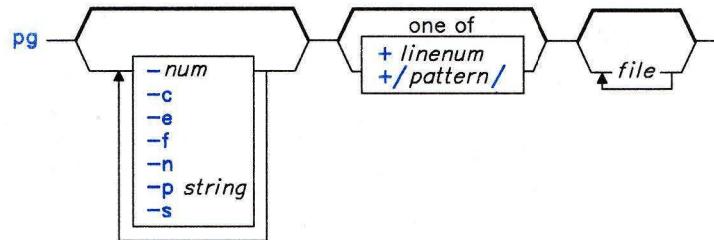
## pg

---

### Purpose

Formats files to the work station.

### Syntax



OL805245

### Description

The **pg** command reads *files* and writes them to standard output one screen at a time. If you specify *file* as - (minus) or run **pg** without arguments, **pg** reads standard input. Each screen is followed by a prompt. If you press the **Enter** key, another page is displayed. The **pg** command lets you back up to review something that has already passed.

To determine work station attributes, **pg** scans the file **terminfo** for the work station type specified by the environment variable **TERM**. The default type is **dumb**. See *AIX Operating System Technical Reference* for information on **terminfo**.

### Subcommands

When **pg** pauses and issues its prompt, you can issue a subcommand. Some of these subcommands change the display to a particular place in the file, some search for specific patterns in the text, and others change the environment in which **pg** works.

The following commands display a selected place in the file:

|              |                                                             |
|--------------|-------------------------------------------------------------|
| <i>page</i>  | Displays the specified <i>page</i> .                        |
| + <i>num</i> | Displays the page <i>num</i> pages after the current page.  |
| - <i>num</i> | Displays the page <i>num</i> pages before the current page. |
| l            | Scrolls the display one line forward.                       |

|               |                                                                                                |
|---------------|------------------------------------------------------------------------------------------------|
| <i>numl</i>   | Displays a screen with the specified line <i>number</i> at the top.                            |
| <b>+numl</b>  | Scrolls the display <i>num</i> lines forward.                                                  |
| <b>-numl</b>  | Scrolls the display <i>num</i> lines backward.                                                 |
| <b>d</b>      | Scrolls half a screen forward. Pressing <b>Ctrl-D</b> also does this.                          |
| <b>-d</b>     | Scrolls half a screen backward. Pressing <b>-Ctrl-D</b> also does this.                        |
| <b>Ctrl-L</b> | Displays the current page again. A single period also does this.                               |
| <b>\$</b>     | Displays the last page in the <i>file</i> . Do not use this when the input is from a pipeline. |

The following commands search for text patterns in the text. You can use the patterns described in “**ed**” on page 371. They must always end with a new-line character, even if the **-n** flag is used. In an expression such as **[a-z]**, the minus means “through” according to the current collating sequence. A collating sequence may define **equivalence classes** for use in character ranges. See “Overview of International Character Support” in *Managing the AIX Operating System* for more information on collating sequences and equivalence classes.

**[num]/pattern/** Search for the *numth* occurrence of *pattern*. The search begins immediately after the current page and continues to the end of the current file, without wraparound. The default for *num* is 1.

**num?pattern?**  
**num^pattern^** Search backward for the *numth* occurrence of *pattern*. The searching begins immediately before the current page and continues to the beginning of the current file, without wraparound. The **^** (circumflex) is useful for the Adds 100 work station, which cannot handle the **?**. The default for *num* is 1.

After searching, **pg** normally displays the line found at the top of the screen. You can change this by adding **m** or **b** to the search command to leave the line found in the middle or at the bottom of the window with all succeeding subcommands. Use the suffix **t** to return to displaying the line with the pattern to the top of the screen.

You can change the **pg** environment with the following subcommands:

|               |                                                                                                                                                                                 |
|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>[num]n</b> | Begins examining the <i>numth</i> next file in the command line. The default <i>num</i> is 1.                                                                                   |
| <b>[num]p</b> | Begins examining the <i>numth</i> previous file on the command line. The default <i>num</i> is 1.                                                                               |
| <b>[num]w</b> | Displays another window of text. If <i>num</i> is present, sets the window size to <i>num</i> .                                                                                 |
| <b>s file</b> | Saves the input in <i>file</i> . Only the current file being examined is saved. This command must always end with a new-line character, even if you specify the <b>-n</b> flag. |

- h** Displays an abbreviated summary of available subcommands.
- q or Q** Quits **pg**.
- !AIX-cmd** Sends the specified AIX command to the shell named in the **SHELL** environment variable. If this is not available, the default shell is used. This command must always end with a new-line character, even if the **-n** flag is used.

At any time when output is being sent to the work station, you can press **QUIT WITH DUMP (Ctrl-V)** or **INTERRUPT (Alt-Pause)**. This causes **pg** to stop sending output and displays the prompt. Then you can enter one of the preceding commands in the normal manner.

**Note:** Some output is lost when when you press **QUIT WITH DUMP (Ctrl-V)** or **INTERRUPT (Alt-Pause)** because any characters waiting in the output queue are purged when the **dQUIT** signal is received.

If standard output is not a work station, **pg** acts like the **cat** command, except that a header is displayed before each file.

While waiting for work station input, **pg** stops running when you press **INTERRUPT (Alt-Pause)**. Between prompts these signals interrupt the current task and place you in the prompt mode.

**Note:** If work station tabs are not set every eight positions, unpredictable results can occur.

## Flags

- c** Moves the cursor to the home position and clears the screen before each page. This flag is ignored if **clear-screen** is not defined for your work station type in the **terminfo** file.
- e** Does not pause at the end of each file.
- f** Does not split lines. Normally, **pg** splits lines longer than the screen width.
- n** Stops processing when a **pg** command letter is entered. Normally, commands must end with a new-line character.
- p string** Uses *string* as the prompt. If the *string* contains a **%d**, the **%d** is replaced by the current page number in the prompt. The default prompt is **:** (colon). If *string* contains spaces, you must quote it.
- s** Highlights all messages and prompts.
- + linenum** Starts at *linenum*.
- num** Specifies the number of lines in the window. On work stations that contain 24 lines, the default is 23.
- + /pattern/** Starts at the first line that contains *pattern*.

## Files

/usr/lib/terminfo/\*  
/tmp/pg\*

## Related Information

The following commands: “**ed**” on page 371 and “**grep**” on page 501.

The **terminfo** file in *AIX Operating System Technical Reference*.

“Overview of International Character Support” in *Managing the AIX Operating System*.

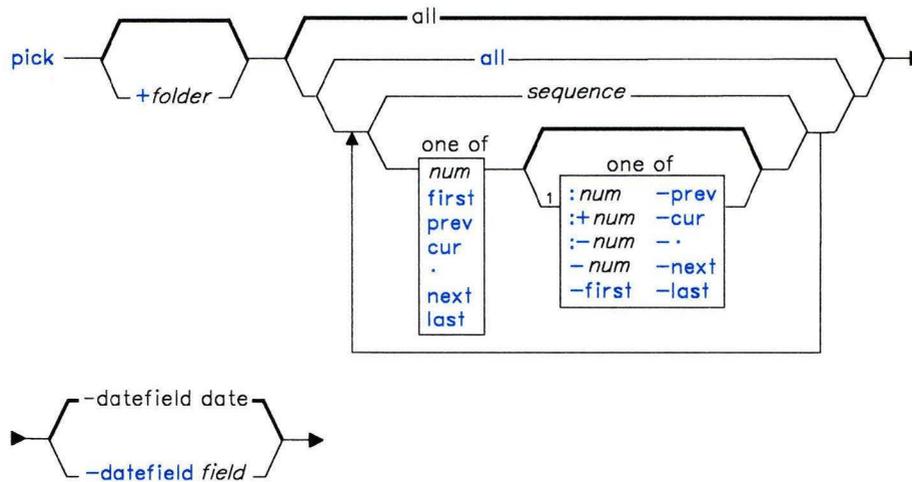
# pick

# pick

## Purpose

Selects messages by content, and creates and modifies sequences.

## Syntax

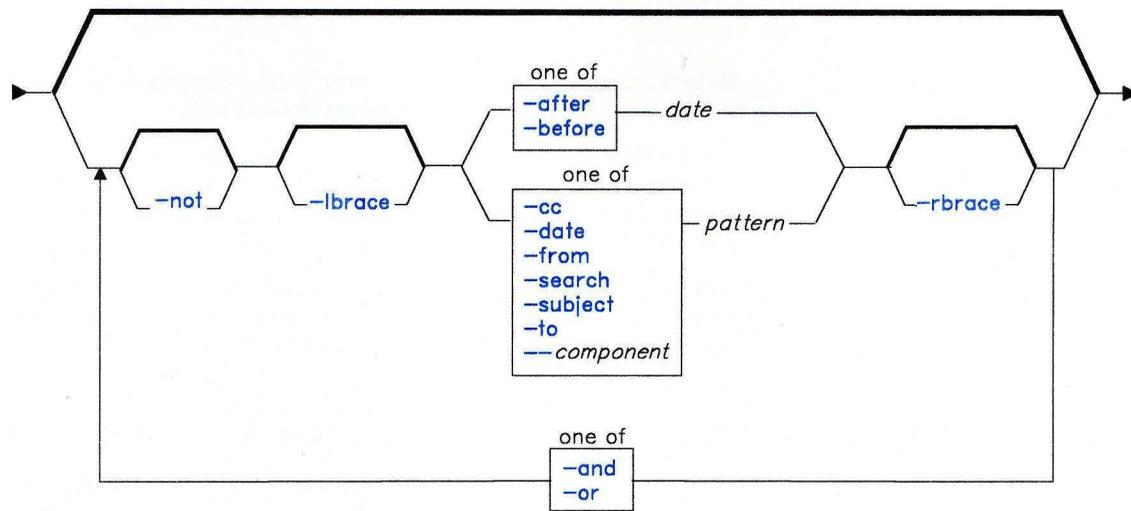


AJ2FL216

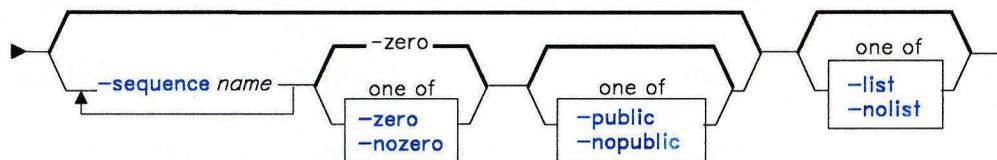
AJ2FL161

<sup>1</sup> Do not put a blank between these items.

OL805308



AJ2FL217



pick — -help —|

AJ2FL253

## Description

The **pick** command is used to select messages and put them into sequences. The **pick** command is part of the MH (Message Handling) package and can be used with other MH and AIX commands.

The **pick** command allows you to select messages that contain particular character patterns or that have particular dates. You can use the **-and**, **-or**, **-not**, **-lbrace**, and **-rbrace** flags to construct compound conditions for selecting messages.

## Flags

- after date** Selects messages with dates later than the specified date. You can use the following date specifications for *date*:
- |                  |                 |                 |
|------------------|-----------------|-----------------|
| <b>yesterday</b> | <b>today</b>    | <b>tomorrow</b> |
| <b>sunday</b>    | <b>monday</b>   | <b>tuesday</b>  |
| <b>wednesday</b> | <b>thursday</b> | <b>friday</b>   |
| <b>saturday</b>  | <i>-dd</i>      | <i>sysdate</i>  |
- The **pick** command treats the days of the week as days in the past. For example, **monday** means last Monday, not today or next Monday. You can use the *-dd* argument to specify a number of days in the past. For example, *-31* means 31 days ago. For *sysdate* you can specify any valid format defined for your system. See “**date**” on page 281 for more information about date formats.
- and** Forms a logical AND operation between two message selecting flags (for example, **pick -after Sunday -and -from mark**). **-and** has precedence over **-or**, but **-not** has precedence over **-and**. Use the **-lbrace** and **-rbrace** flags to override this precedence.
- before date** Selects messages with dates earlier than the specified date. See the **-after** flag for how to specify *date*.
- cc pattern** Selects messages that contain the character string *pattern* in the **Cc:** field.
- date pattern** Selects messages that contain the character string *pattern* in the **Date:** field.
- datefield field** Specifies which dated field is parsed when the **-after** and **-before** flags are given. By default, **pick** uses the **Date:** field.
- +folder msgs** Specifies the messages that you want to search. *msgs* can be several messages, a range of messages, or a single message. You can use the following message references when specifying *msgs*:
- |             |              |                 |
|-------------|--------------|-----------------|
| <i>num</i>  | <b>first</b> | <b>prev</b>     |
| <b>cur</b>  | <b>.</b>     | <b>next</b>     |
| <b>last</b> | <b>all</b>   | <i>sequence</i> |
- The default for *msgs* is **all**. The default folder is the current folder. If you specify a folder, that folder becomes the current folder.
- from pattern** Selects messages that contain the character string *pattern* in the **From:** field.
- help** Displays help information for the command.

|                               |                                                                                                                                                                                                                                                                                                                                                                                  |
|-------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-lbrace</b>                | Groups <b>-and</b> , <b>-or</b> , and <b>-not</b> operations. Operations between the <b>-lbrace</b> and <b>-rbrace</b> flags are evaluated as one operation. You can nest the <b>-lbrace</b> and <b>-rbrace</b> flags.                                                                                                                                                           |
| <b>-list</b>                  | Sends a list of selected message numbers to standard output. This allows you to use the <b>pick</b> command to generate message numbers to use as input for other commands. For example, <code>scan 'pick -after tuesday -list'</code> scans all messages in the current folder that were sent after last Tuesday. <b>-list</b> is the default if no sequence is specified.      |
| <b>-nolist</b>                | Keeps the <b>pick</b> command from generating a list of the selected message numbers (see the <b>-list</b> flag). If a sequence is specified, <b>-nolist</b> is the default.                                                                                                                                                                                                     |
| <b>-npublic</b>               | Restricts the specified sequence to your usage. <b>-npublic</b> does not restrict the messages in the sequence, only the sequence. This option is the default if the folder is write-protected from other users.                                                                                                                                                                 |
| <b>-not</b>                   | Forms a logical NOT operation on a message selecting flag (for example, <code>pick -not -from george</code> ). This construction evaluates to all messages not chosen by the message selecting flag. <b>-not</b> has precedence over <b>-and</b> , and <b>-and</b> has precedence over <b>-or</b> . Use the <b>-lbrace</b> and <b>-rbrace</b> flags to override this precedence. |
| <b>-nozero</b>                | Appends the selected messages to the specified sequence (see the <b>-zero</b> flag).                                                                                                                                                                                                                                                                                             |
| <b>-or</b>                    | Forms a logical OR operation on two message-selecting flags (for example, <code>pick -from amy -or -from mark</code> ). <b>-not</b> has precedence over <b>-and</b> , and <b>-and</b> has precedence over <b>-or</b> . Use the <b>-lbrace</b> and <b>-rbrace</b> flags to override this precedence.                                                                              |
| <b>-public</b>                | Makes the specified sequence available to other users. <b>-public</b> does not make protected messages available, only the sequence. This option is the default if the folder is not write-protected from other users.                                                                                                                                                           |
| <b>-rbrace</b>                | Groups <b>-and</b> , <b>-or</b> , and <b>-not</b> operations. Operations between the <b>-lbrace</b> and <b>-rbrace</b> flags are evaluated as one operation. You can nest the <b>-lbrace</b> and <b>-rbrace</b> flags.                                                                                                                                                           |
| <b>-search <i>pattern</i></b> | Selects messages that contain the character string <i>pattern</i> anywhere in the message.                                                                                                                                                                                                                                                                                       |
| <b>-sequence <i>name</i></b>  | Stores the messages selected by the <b>pick</b> command in the sequence <i>name</i> .                                                                                                                                                                                                                                                                                            |
| <b>-to <i>pattern</i></b>     | Selects messages that contain the character string <i>pattern</i> in the <b>To:</b> field.                                                                                                                                                                                                                                                                                       |

## pick

---

- zero** Clears the specified sequence before placing the selected messages into the sequence. This flag is the default (see the **-nozero** flag).
- component *pattern*** Selects messages that contain the character string *pattern* in the heading field *component* (for example, `pick --reply-to amy`).

## Profile Entries

- Current-Folder:** Sets your default current folder.  
**Path:** Specifies your *user-mh-directory*.

## Files

- `$HOME/.mh-profile` The MH user profile.

## Related Information

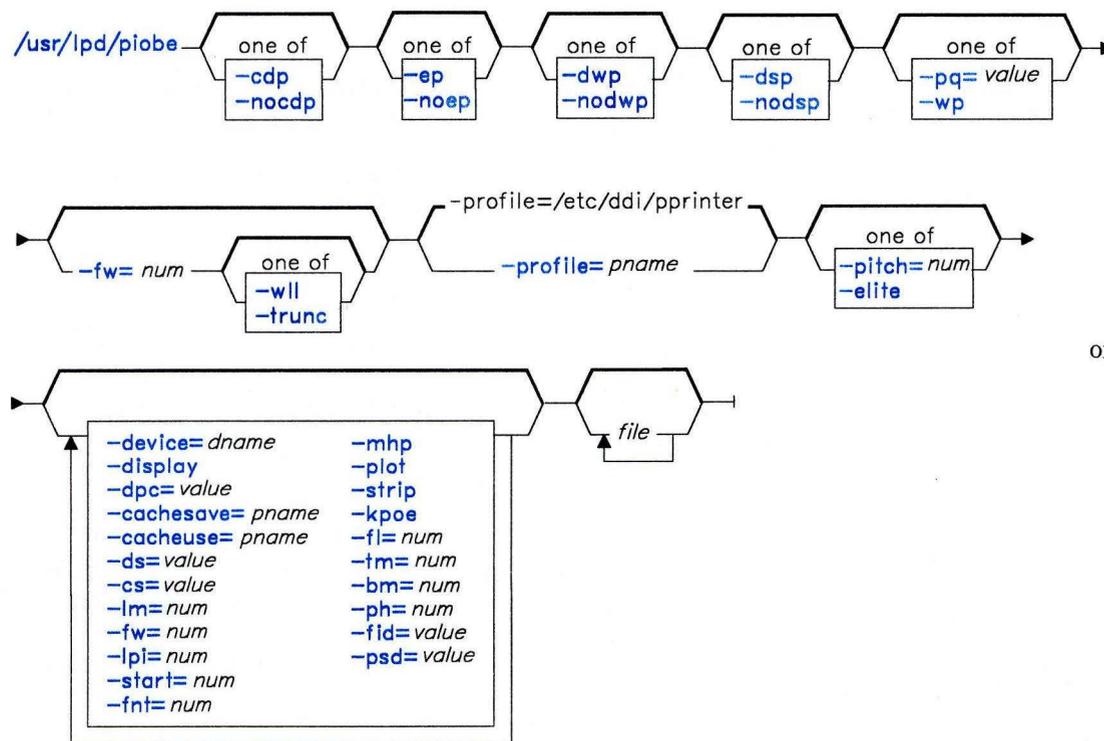
- The MH command “**mark**” on page 637.
- The **mh-profile** file in *AIX Operating System Technical Reference*.
- “Overview of the Message Handling Package” in *Managing the AIX Operating System*.

# pio-be

## Purpose

Writes a file to standard output in a format suitable for sending to a line printer.

## Syntax



OL805391

OL805452

## Description

The **pio-be** command writes *file* to its standard output in a form that is suitable for a line printer. If you do not specify a *file*, **pio-be** reads standard input. **pio-be** is normally called by the **qdaemon** command after you have enqueued a file with the **print** command (see “**print**” on page 767). The **qdaemon** directs the output from **pio-be** to the appropriate device.

## Flags

You can specify the following flags on the **print** command line or in the **/etc/qconfig** file (see *AIX Operating System Technical Reference*).

- bm = num** Sets the bottom margin to *num* lines from the top of the page.
- cachesave = pname** Specifies the path name of the file where the font Personal PagePrinter adapter is to be saved after the input data stream prints.
- cacheuse = pname** Specifies the path name of the font cache file loaded in the IBM Personal PagePrinter adapter before the input data stream is printed.
- cdp**  
**-nocdp** Turns the condensed printing mode on (**-cdp**) or off (**-nocdp**).
- cs = value** Uses PC code set 1 or 2.
- device = dname** Specifies the name of a printer stanza in the printer configuration file (see "Files" on page 756).
- display** Specifies that the input data stream has KSR code page controls.
- dpc = value** Prints in the specified color. Valid color *values* are **red**, **blue**, **yellow**, and **black**.
- ds = value** Specifies how the input data stream is to be printed on the IBM 4216 Personal PagePrinter. Valid values are the following:  
**ps ps**  
**ppxl** Proprinter XL emulation  
**any** This is the default.  
**ps** is assumed if the first character of the data stream is %. Otherwise, Proprinter XL emulation is assumed.
- dsp**  
**-nodsp** Turns the double strike mode on (**-dsp**) or off (**-nodsp**).
- dwp**  
**-nodwp** Turns the double wide printing mode on (**-dwp**) or off (**-nodwp**).
- elite** Sets the character pitch to **12**, the same as specifying **-pitch = 12**.
- ep**  
**-noep** Turns the emphasized printing mode on (**-ep**) or off (**-noep**).
- fid = value** Specifies the font identifier for an IBM 5202 Quietwriter® III Printer font. Valid values for embedded fonts are **11** (Courier 10), **85** (Courier 12), **254** (Courier 17), and **159** (Boldface).

|                         |                                                                                                                                                                                                                                                                                                                       |
|-------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                         | Values for fonts in the pluggable cartridges precede the font name on the cartridge label.                                                                                                                                                                                                                            |
| <b>-fl = num</b>        | Sets the form length to <i>num</i> .                                                                                                                                                                                                                                                                                  |
| <b>-fnt = num</b>       | Allows font change. Valid values for <i>num</i> are 1 through 8.                                                                                                                                                                                                                                                      |
| <b>-fw = num</b>        | Sets the right margin at <i>num</i> characters from the left edge of the carriage.                                                                                                                                                                                                                                    |
| <b>-kpoe</b>            | Forgives keying mistakes and ignores invalid flags. If you specify this flag, <b>piobe</b> processes the job and sends you no message. If you do not specify this flag, <b>piobe</b> does not forgive invalid flags and does not print the job. In this case, it sends you a message detailing the error.             |
| <b>-lm = num</b>        | Sets the left margin at <i>num</i> characters from the left edge of the carriage.                                                                                                                                                                                                                                     |
| <b>-lpi = num</b>       | Sets the number of lines per inch to <i>num</i> . Valid settings are 6 and 8.                                                                                                                                                                                                                                         |
| <b>-mhp</b>             | Allows the horizontal position on the print line to be maintained for line feed and vertical tab controls, if desired.                                                                                                                                                                                                |
| <b>-ph = num</b>        | Allows you to use single-sheet paper in the Quietwriter printer. The printer stops at the end of each page, beeps three times, and waits for you to push the start button. <i>num</i> can have the following values:<br><b>0</b> Manual operation.<br><b>1</b> Sheetfeed operation.<br><b>2</b> Continuous operation. |
| <b>-pitch = num</b>     | Sets the character pitch to <i>num</i> .                                                                                                                                                                                                                                                                              |
| <b>-plot</b>            | Specifies that the input data is to be passed through without modification. This allows arbitrary files to be printed on arbitrary printers.                                                                                                                                                                          |
| <b>- = value</b>        | Specifies the printer name. Messages that report a printer is out of paper or needs attention include the name of the printer. The default name of the printer is specified in the <b>/etc/qconfig</b> file.                                                                                                          |
| <b>-pq = value</b>      | Prints in specified print quality. Valid quality <i>values</i> are <b>dp</b> , <b>text</b> , and <b>letter</b> .                                                                                                                                                                                                      |
| <b>-profile = pname</b> | Specifies the name of a printer configuration file. The default name is <b>/etc/ddi/pprinter</b> .                                                                                                                                                                                                                    |

## piobe

---

|                            |                                                                                                                                                                                  |
|----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-psd</b> = <i>value</i> | Specifies a paper source drawer for the optional IBM 5202 Quietwriter III Printer two-drawer sheetfeeder. Valid values are 1 (top drawer), 2 (bottom drawer), and 3 (envelopes). |
| <b>-start</b> = <i>num</i> | Sets the starting page number to <i>num</i> .                                                                                                                                    |
| <b>-strip</b>              | Strips all multibyte controls from the data stream. This flag is useful in filter mode in order to send data that has imbedded printer controls to a nonprinter device.          |
| <b>-tm</b> = <i>num</i>    | Sets the top margin to <i>num</i> lines.                                                                                                                                         |
| <b>-trunc</b>              | Specifies that lines exceeding the value set by <b>-fw</b> should be truncated.                                                                                                  |
| <b>-wll</b>                | Specifies that lines exceeding the value set with the <b>-fw</b> flag should overflow to the next line. This is the reverse of the <b>-trunc</b> flag.                           |
| <b>-wp</b>                 | Selects word processing mode, the same as specifying <b>-pq = letter</b> .                                                                                                       |

## Files

|                          |                                     |
|--------------------------|-------------------------------------|
| <i>/etc/ddi/pprinter</i> | Parallel configuration information. |
| <i>/etc/ddi/sprinter</i> | Serial configuration information.   |

## Related Information

The following commands: “**print**” on page 767 and “**qdaemon**” on page 802.

The **qconfig** file in *AIX Operating System Technical Reference*.

---

## **portmap**

---

### **Purpose**

Maps RPC programs to the servicing ports on RPC servers

### **Syntax**

`portmap` —|

A5AC5022

### **Description**

The **portmap** program, also known as the **portmapper**, maps RPC program number to the port numbers on RPC servers in order for NFS clients to make RPC service calls. When an RPC server starts, it contacts the **portmap** program to register the RPC programs it is prepared to serve and the port on which it is listening for calls. When NFS clients call an RPC procedure for a given program number, the clients contact the **portmap** program to determine the port number to which they should send the packets.

The **portmap** program is usually started from an entry in the **inetd.conf** file.

---

#### **Japanese Language Support Information**

If Japanese Language Support is installed on your system, this command is not available.

---

### **Files**

/etc/portmap  
/etc/rc.tcpip

### **Related Information**

The following command: “**rpcinfo**” on page 845.

The section on NFS in *Managing the AIX Operating System*.

# post

---

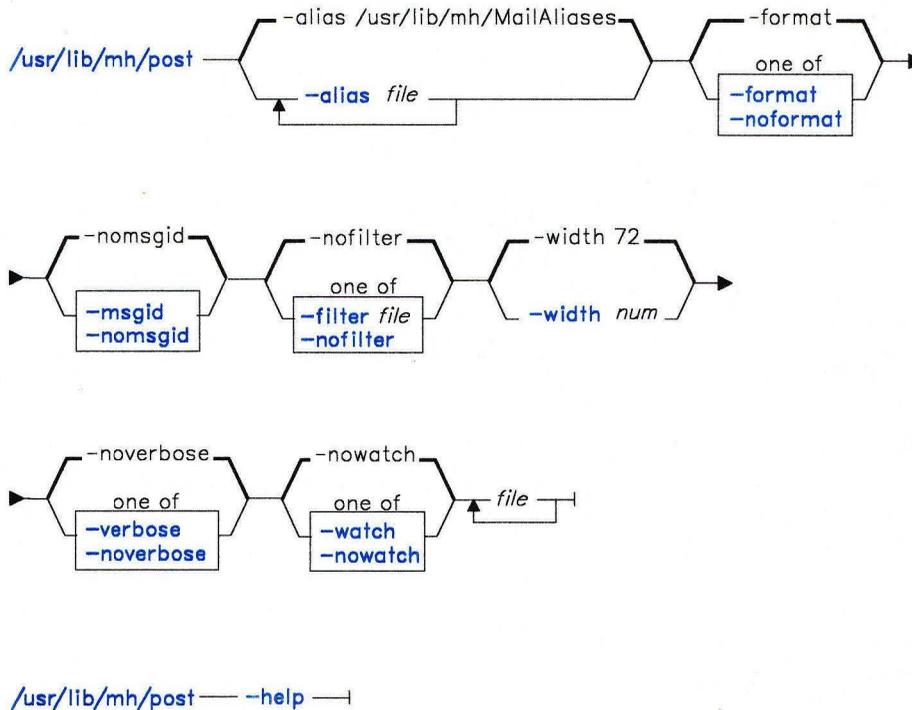
## post

---

### Purpose

Delivers a message.

### Syntax



AJ2FL233

### Description

The **post** command is used to route messages to the proper destinations. **post** is not designed to be run directly by the user; it is designed to be called by other programs. The **post** command is part of the MH (Message Handling) package.

The **post** command searches all components of a message that specify a recipient's address and parses each address to check for proper format. **post** puts proper addresses in the

---

standard format and calls the **sendmail** command. **post** also performs header operations, such as appending the **Date:** and **From:** components and processing the **Bcc:** component.

The **post** command may report errors when parsing complex addresses (for example, @A:harold@B.UUCP). If the **sendmail** program is installed on your system and you use complex addresses, use the **spost** command instead of the **post** command.

## Flags

- alias file** Searches the specified mail alias file for addresses. You can use this flag repetitively to specify multiple mail alias files. **post** automatically searches the file `/usr/lib/mh/MailAliases`.
- filter file** Uses the header components in the specified file for copies of the message sent to **Bcc:** recipients.
- format** Puts all recipient addresses in a standard format for the delivery transport system. This flag is the default.
- help** Displays help information for the command.
- msgid** Adds a message-identification component (such as **Message-ID:**) to the message.
- nofilter** Strips the **Bcc:** header from the message for the **To:** and **cc:** recipients. Sends the message with minimal headers to the **Bcc:** recipients. This flag is the default.
- noformat** Does not alter the format of the recipient addresses.
- nomsgid** Does not add a message-identification component to the message. This flag is the default.
- noverbose** Does not display information during the delivery of the message to the **sendmail** command. This flag is the default.
- nowatch** Does not display information during delivery by the **sendmail** command. This flag is the default.
- verbose** Displays information during the delivery of the message to the **sendmail** command. This information allows you to monitor the steps involved.
- watch** Displays information during the delivery of the message by the **sendmail** command. This information allows you to monitor the steps involved.
- width num** Sets the width of components that contain addresses. The default is 72 columns.

## post

---

### Files

/usr/lib/mh/MailAliases  
/usr/lib/mh/mtstailor

The default mail alias file.  
The MH tailor file.

### Related Information

The following commands: “**ali**” on page 48, “**conflict**” on page 196, “**mhmail**” on page 646, “**send**” on page 893, “**sendmail**” on page 897, “**spost**” on page 978, and “**whom**” on page 1222.

The **mh-alias**, **mh-format**, **mh-mail**, and **mh-profile** files in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

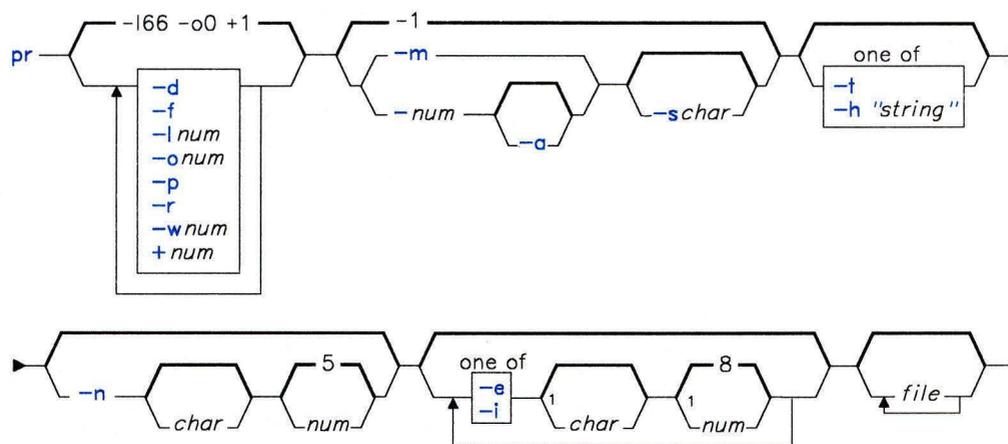
---

**pr**


---

**Purpose**

Writes a file to standard output.

**Syntax**


<sup>1</sup> Do not put a blank between these items.

OL805437

**Description**

The `pr` command writes *file* to the standard output. If you do not specify *file* or if *file* is a `-` (minus), `pr` reads standard input. A heading that contains the page number, date, time, and the name of the file separates the output into pages.

Unless specified, columns are of equal width and separated by at least one space. Lines that are too long for the page width are cut off. If the standard output is a work station, `pr` does not display any error messages until it has ended.

**Flags**

- `-a`            Displays multicolumn output across the page.
- `-d`            Double-spaces the output.

- e[*char*][*num*]** Expands tabs to character positions  $num + 1$ ,  $2 * num + 1$ ,  $3 * num + 1$ , and so on. The default value of *num* is 8. Tab characters in the input expand to the appropriate number of spaces to line up with the next tab setting. If you specify *char* (any character other than a digit) that character becomes the input tab character. The default value of *char* is the ASCII TAB character.
- f** Uses a form feed character to advance to a new page. (Otherwise **pr** issues a sequence of line feed characters.) Pauses before beginning the first page if the standard output is a work station.
- h "*string*"** Displays *string* as the page header instead of the file name. The flag and string should be separated by a blank.
- i[*char*][*num*]** In the *output*, replaces white space wherever possible by inserting tabs to character positions  $num + 1$ ,  $2 * num + 1$ ,  $3 * num + 1$ , and so on. The default value of *num* is 8. If you specify *char* (any character other than a digit), that character becomes the output tab character. (The default value of *char* is the ASCII TAB character).
- l*num*** Sets the length of a page to *num* lines (the default is 66).
- m** Combines and writes all files at the same time, with each file in a separate column. (This overrides the *-num* and *-a* flags).
- n[*char*][*num*]** Provides *num*-digit line numbering (the default value of *num* is 5). The number occupies the first *num* + 1 character positions of each column of normal output or each line of *-m* output. If you specify *char* (any character other than a digit), that character is added to the line number to separate it from whatever follows (the default value of *char* is an ASCII TAB character).
- o*num*** Indents each line by *num* character positions (the default is 0). The number of character positions per line is the sum of the width and offset.
- p** Pauses before beginning each page if the output is directed to a work station. (**pr** sounds the alarm at the work station and waits for you to press the **Enter** key.)
- r** Does not display diagnostic messages if the system cannot open files.
- s*char*** Separates columns by the single character *char* instead of by the appropriate number of spaces (the default for *char* is an ASCII TAB character).
- t** Does not display the five-line identifying header and the five-line footer. Stops after the last line of each file without spacing to the end of the page.
- w*num*** Sets the width of a line to *num* character positions (the default value is 72 for equal-width multicolumn output, no limit otherwise).

- `-num`            Produce *num*-column output (the default is 1). The `-e` and `-i` flags are assumed for multicolumn output.
- `+num`            Begin the display with page *num* (the default value is 1).

## Examples

1. To print a file with headings and page numbers on the printer:

```
pr prog.c | print
```

This adds page headings to `prog.c` and sends it to the `print` command. The heading consists of the date the file was last modified, the file name, and the page number.

2. To specify a title:

```
pr -h "MAIN PROGRAM" prog.c | print
```

This prints `prog.c` with the title `MAIN PROGRAM` in place of the file name. The modification date and page number are still printed.

3. To print a file in multiple columns:

```
pr -3 word.lst | print
```

This prints the file `word.lst` in three vertical columns.

4. To print several files side by side on the paper:

```
pr -m -h "Members and Visitors" member.lst visitor.lst | print
```

This prints `member.lst` and `visitor.lst` side by side with the title `Members and Visitors`.

5. To modify a file for later use:

```
pr -t -e prog.c > prog.notab.c
```

This replaces tab characters in `prog.c` with blanks and puts the result in `prog.notab.c`. Tab positions are at columns 9, 17, 25, 33, . . . . The `-e` tells `pr` to replace the tab characters; the `-t` suppresses the page headings.

**pr**

---

## **Files**

`/dev/tty*` To suspend messages.

## **Related Information**

The following command: “**cat**” on page 137.

---

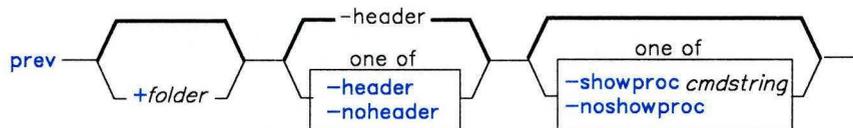
## prev

---

### Purpose

Shows the previous message.

### Syntax



prev — -help —|

AJ2FL162

### Description

The **prev** command is used to display the previous message in a folder. **prev** is equivalent to the **show** command with **prev** specified as the message. **prev** is part of the MH (Message Handling) package and can be used with other MH and AIX commands.

The **prev** command links to the **show** program and passes **show** its flags and attributes.

**Note:** If you link to **prev** and call that link something other than **prev**, your link will function like the **show** command, rather than like the **prev** command.

The **show** command invokes a program to perform the listing. The system default is **/bin/pg**. You can define your own default with the **showproc:** entry in **\$HOME/.mh-profile**. If you set **showproc:** entry to **mhl**, **show** calls an internal **mhl** routine instead of the **mhl** command. You can also specify the program to perform a listing in the *cmdstring* of the **-showproc** flag.

The **show** command passes any flags that it does not recognize to the program performing the listing. Thus, you can specify flags for the listing program, as well as the flags described in this command section.

## Flags

|                                  |                                                                                                                              |
|----------------------------------|------------------------------------------------------------------------------------------------------------------------------|
| <code>+folder</code>             | Specifies the folder that contains the message you want to show.                                                             |
| <code>-header</code>             | Displays a one-line description of the message being shown. The description includes the folder name and the message number. |
| <code>-help</code>               | Displays help information for the command.                                                                                   |
| <code>-noheader</code>           | Does not display a one-line description of each message being shown.                                                         |
| <code>-noshowproc</code>         | Uses <code>/bin/cat</code> to perform the listing.                                                                           |
| <code>-showproc cmdstring</code> | Uses the specified command string to perform the listing.                                                                    |

## Profile Entries

|                        |                                                 |
|------------------------|-------------------------------------------------|
| <b>Current-Folder:</b> | Sets your default current folder.               |
| <b>Path:</b>           | Specifies your <code>user_mh_directory</code> . |
| <b>showproc:</b>       | Specifies the program used to show messages.    |

## Files

|                                 |                      |
|---------------------------------|----------------------|
| <code>\$HOME/.mh-profile</code> | The MH user profile. |
|---------------------------------|----------------------|

## Related Information

Other MH commands: “**next**” on page 694, “**show**” on page 942.

The **mh-format**, **mh-mail**, and **mh-profile** files in *AIX Operating System Technical Reference*.

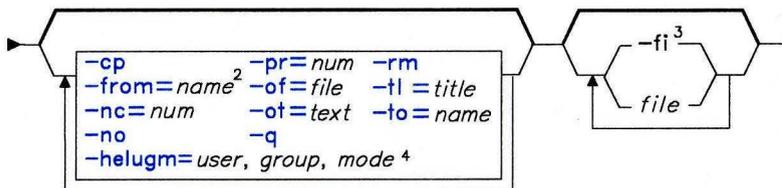
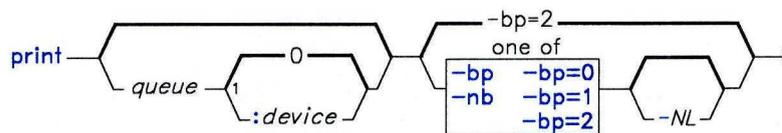
“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

# print

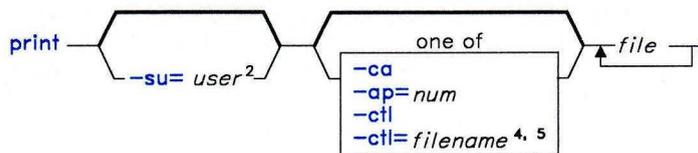
## Purpose

Enqueues a file.

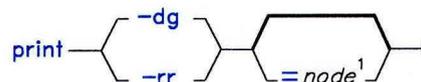
## Syntax



OL805348



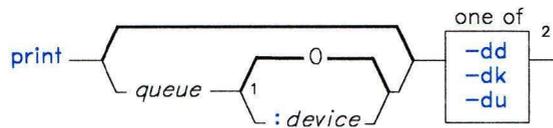
OL805147



OL805354

# print

---



<sup>1</sup> Do not put a blank between these items.

<sup>2</sup> Only members of the system group can use these flags.

<sup>3</sup> Only specify `-fi` once.

OL805328

## Description

The **print** command is a general purpose utility for enqueueing requests to a shared resource, typically a printer device. With the **print** command you can do the following:

- Enqueue print requests
- Cancel print requests
- Alter the priority of print requests
- Display the status of print queues and devices
- Specify the printing of security labels on hard copy.

For complete information about security labels, see the discussion of printer security labels in *Managing the AIX Operating System*.

To enqueue files on a specific queue, specify `-queue`. If more than one device services a queue, you can also request a particular device by specifying `:device` after the name of the queue. If you do not specify a device, the job is sent to the first available device. If you do not specify a *file*, **print** copies standard input into a file and enqueuees it for printing.

Print requests can have operator messages associated with them. This feature is useful in a distributed services environment. You can use messages to tell the user operating the printer to load a special form into the printer before allowing this job to print. These messages are specified with the `-of` and `-ot` flags. When **qdaemon**, the daemon that processes print requests, is ready to begin a request that has an associated message, the system displays the message on the console of the machine where **qdaemon** is running. The text of the message is accompanied by a prompt that tells the printer operator how to signal the request to continue or cancel the request.

**Notes:**

1. Before you can print a file, you must have read access to it. To remove a file, you must also have write access to the directory that contains the file.
2. If you want to continue changing the *file* after you issue the **print** command but before it is printed, you must use the **-cp** flag.
3. The operator must respond to messages sent with the **-of** and **-ot** flags. Print requests with these flags stop processing until the operator responds.
4. When enqueueing files on a printer, flags and file names can be interspersed in any order.
5. Blanks between flags and their arguments are not permitted.

## Flags

If you give **print** a list of *file* names, it enqueues them all for printing on the default printer.

**-ap = num** Changes to *num* the priority of the named file. The file must have been submitted for printing prior to entering the **print** command with this flag. See “-pr” on page 770 for a description of priorities.

**-bp = num**  
**-bp**  
**-nb**

Controls the printing of burst pages according to the value of *num* as follows:

- 0** Does not print headers or trailers.
- 1** Prints one header page before each *file*. No trailer appears.
- 2** Prints a header page at the beginning and a trailer page at the end of each *file*.

The **header** stanza in the **qconfig** file defines the default treatment of burst pages.

Specifying only **-bp** is the same as specifying **-bp = 2**. Specifying **-nb** is the same as specifying **-bp = 0**.

**-ca**

Cancels the printing of the named *files*. *STDIN. ?# use get ex*

**-ctl[ =filename]**

Specifies security labels and access codes for a file located on a remote system. You must be a superuser to use this flag.

**-cp**

Copies the file. Ordinarily, to save disk space, **print** remembers the name of the file, but does not actually copy the file itself. Use the **-cp** flag if you want to continue changing the file while you are waiting for the current copy to be printed.

## print

---

- fi** Causes **print** to act as a filter. The **print** command automatically reads standard input if you do not specify *files* as arguments. However, if you specify *file* arguments, you can use the **-fi** flag to force **print** to read standard input at the appropriate time.
- nc = num** Prints *num* copies of the file. Normally a file prints only once.
- nl** Suppresses top-of-page and bottom-of-page output labeling.
- no** Notifies you when your job is finished. If the **-to** flag is also used, **print** notifies the user for whom the request is intended (see the **-to** flag on page 770).
- of = file** Submits an operator message with a print request. The specified *file* contains the text of the message.
- ot = text** Submits an operator message with a print request. The specified *text* contains the text of the message.
- pr = num** Sets the priority of the named *file* to *num*. Higher numbers assign higher priority. The default priority is 15. The maximum priority is 20 for most users and 30 for the users with superuser authority and members of the system group (group 0).
- q** Displays the status of the queues and printers. You can specify an *argname* to view a single queue. The environment variable **NLTIME** controls the appearance of the **time** field.
- rm** Removes the file after it prints.
- tl = title** Puts *title* on the header page and displays it when the **-q** flag is specified. Normally the job title is the name of the file. If **print** reads from standard input, the job title is **STDIN.#** where # is the process ID of the **print** command.
- to = name** Labels the output for delivery to *name*. Normally the output is labeled for delivery to the person issuing the print request.

In addition to the previous flags that are available to all users, the **print** command accepts the following flags when they are entered by users who have superuser authority or users who are members of the system group:

- dd** Turns off the device associated with *queue*. The **qdaemon** no longer sends jobs to the device, and entering **print -q** shows its status as OFF. Any job currently running on the device is allowed to finish.
- dg[ = node]** Kills the **qdaemon** after all currently running jobs are finished. Use of this flag is the only clean way to bring the **qdaemon** down. Use of the **kill** command may cause problems, such as jobs hanging up in the queue. If the **qdaemon** is on a remote node, specify the node. You can specify it as either a node ID or nickname. If no node is specified, the local node is assumed.

- dk** Acts the same as **-dd**, except current jobs are killed. They remain in the queue, and are run again when the device is turned on.
- du** Turns on the device associated with *queue*. The **qdaemon** sends jobs to it again and entering `print -q` shows its status as **READY**.
- Note:** If more than one device is associated with a queue, you must specify the device as well as the queue when you use the **-dd**, **-dk**, and **-du** flags. Devices are numbered, starting at zero, in the order that they appear in the **qconfig** file. For example, `-lp:0` designates the first device on the **lp** queue. `-lp` designates the same device only if there is no other device on that queue.
- from = name** Labels the output as though *name* had submitted it. You can only use this flag with superuser authority.
- rr[ = node]** Forces the **qdaemon** to reread the **qconfig** file after all currently running jobs are finished. With this flag, a user with superuser authority can change the **qconfig** file without having to kill and restart the **qdaemon**. If the **qdaemon** is on a remote node, specify the node either a node ID or nickname. If no node is specified, the local node is assumed.
- su = user** Cancels or changes the priority on another *user's* job when used with the **-ca** or the **-ap** flags. For example, a job report submitted by user `ann` can be cancelled as follows:
- ```
print -su=ann -ca report
```

The **print** command passes flags it does not recognize to the backend that does the printing. Thus, for each queue there are flags not described above that can be included on the **print** command line. See “**piobe**” on page 753 for a list of these flags.

## Examples

1. To print a file on the default printer:

```
print memo
```

2. To print a file with page numbers:

```
pr prog.c | print
```

The **pr** command puts a heading at the top of each page that includes the date the file was last modified, the name of the file, and the page number. The **print** command then prints the file.

3. To see if a file is still waiting to be printed:

```
print -q
```

## print

---

This command displays the status of the queues and printers. If a file has not yet printed, then it appears in the queue status listing. If you piped data to **print**, as in Example 2, then it is listed as PRIMARY.OUTPUT.

4. To stop printing a file:

```
print -ca chapter1
```

This command cancels the request you made earlier to print the file `chapter1`. If the file is currently being printed, then the printer stops immediately. If the file has not yet printed, then it is removed from the queue so that it will not be printed. If the file is not in the queue, **print** displays the message:

```
no such request from you -- perhaps it's done?
```

5. To disconnect a printer from the queueing system:

```
print -a:2 -dd
```

This command stops print requests from being sent to the third printer that serves the `-a` queue. If a file is currently being printed, it is allowed to finish. You must be a member of the **system** group (group 0) to run this command.

**Note:** The printers serving a given queue are numbered starting with zero in the order that they appear in the `/etc/qconfig` file.

## Files

<code>/etc/qdaemon</code>	Queueing daemon.
<code>/usr/lpd/qdir/*</code>	Queue requests.
<code>/usr/lpd/stat/*</code>	Information on the status of the devices.
<code>/usr/spool/qdaemon/*</code>	Temporary copies of enqueued files.
<code>/etc/qconfig</code>	Queue configuration file.
<code>/etc/security/config</code>	Specifies output labelling information.

## Related Information

The following commands: “**piobe**” on page 753, “**pr**” on page 761, and “**qdaemon**” on page 802.

The **qconfig** file in *AIX Operating System Technical Reference*.

The discussion of the printer subsystem in *Managing the AIX Operating System*.

“Overview of International Character Support” in *Managing the AIX Operating System*.

---

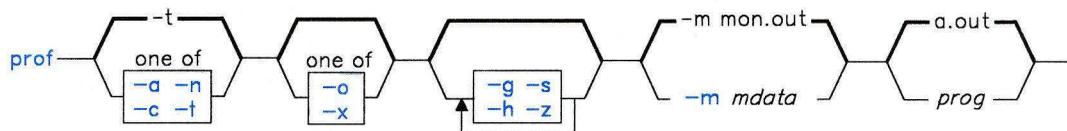
# prof

---

## Purpose

Displays program profile data.

## Syntax



OL805438

## Description

The **prof** command interprets profile data collected by the **monitor** subroutine for the object file **prog** (**a.out** by default). It reads the symbol table in the object file *prog* and correlates it with the profile file (**mon.out** by default). **prof** displays, for each external text symbol, the percentage of execution time spent between the address of that symbol and the address of the next, the number of times that function was called, and the average number of milliseconds per call.

For the number of calls to a function to be tallied, you must have compiled the file using the **-p** flag of the **cc** command. This flag also arranges for the object file to include a special profiling startup function that calls the **monitor** subroutine at the beginning and end of execution. It is the call to **monitor** at the end of execution that writes **mon.out**. Thus, only programs that explicitly **exit** or **return** from **main** cause the **mon.out** file to be produced.

**Note:** No more than 600 functions can have call counters established during program execution. If you exceed this limit, **prof** overwrites other data and damages the **mon.out** file. **prof** automatically reports the number of call counters used whenever the number exceeds 500.

---

### Japanese Language Support Information

This command has not been modified to support Japanese characters.

---

## Flags

The mutually exclusive flags **a**, **c**, **n**, and **t** determine how **prof** sorts the output lines:

- a** Sorts by increasing symbol address.
- c** Sorts by decreasing number of calls.
- n** Sorts lexically by symbol name.
- t** Sorts by decreasing percentage of total time (default).

The mutually exclusive flags **o** and **x** specify how to display the address of each symbol monitored.

- o** Displays each address in octal, along with the symbol name.
- x** Displays each address in hexadecimal, along with the symbol name.

Use the following flags in any combination:

- g** Includes nonglobal symbols (static functions). This option requires object code that was compiled with the **-g** flag.
- h** Suppresses the heading normally displayed on the report. (This is useful if the report is to be processed further.)
- m *mdata*** Takes profiling data from *mdata* instead of **mon.out**.
- s** Displays a summary of monitoring parameters and statistics on standard error.
- z** Includes all symbols in the profile range, even if associated with zero calls and zero time.

## Files

- mon.out** Default profile.
- a.out** Default object file.

## Related Information

The following commands: “**cc**” on page 140 and “**nm**” on page 705.

The **exit** and **profil** system calls and the **monitor** subroutine in *AIX Operating System Technical Reference*.

---

# profiler

---

## Purpose

Profiles the operating system.

## Syntax

`prfld` */unix*  
*kernel-image*

`prfstat` *on*  
*off*

`prfdc` *file* *minutes* *hour*

`prfsnap` *file*

`prfpr` *file* *cutoff* */unix*  
*kernel-image*

OL805006

## Description

With the **prfld**, **prfstat**, **prfdc**, **prfsnap**, and **prfpr** commands, you can examine the activity of the AIX Operating System.

## profiler

---

### **prfld**

Use **prfld** to initialize the recording mechanism in the system. It produces a table containing the starting address of each system subroutine as extracted from *kernel-image*.

---

### **Japanese Language Support Information**

This command has not been modified to support Japanese characters.

---

### **prfstat**

Use **prfstat** to enable or disable the sampling mechanism. **prfstat** also reveals the number of text addresses being measured.

---

### **Japanese Language Support Information**

This command has not been modified to support Japanese characters.

---

### **prfdc, prfsnap**

Use **prfdc** and **prfsnap** to collect profiler data by copying the current value of all the text address counters to a file where the data can be analyzed. **prfdc** stores the counters into *file* every specified *minutes* and turns off at *hour* (0-24). **prfsnap** collects data at the time of invocation only, adding the counter values to *file*.

---

### **Japanese Language Support Information**

This command has not been modified to support Japanese characters.

---

### **prfpr**

Use **prfpr** to format the data collected by **prfdc** or **prfsnap**. It converts each text address to the nearest text symbol (as found in *kernel-image*) and displays it if the percent activity for that range is greater than *cutoff*.

---

### **Japanese Language Support Information**

This command has not been modified to support Japanese characters.

---

## **Files**

<code>/dev/prf</code>	Interface to profile data and text addresses.
<code>/unix</code>	System kernel image file.

## **Related Information**

The `prf` file in *AIX Operating System Technical Reference*.

# prompter

---

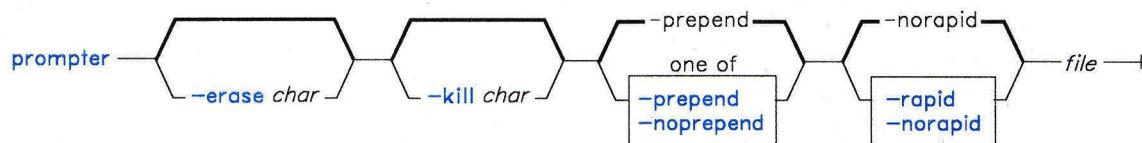
## prompter

---

### Purpose

Invokes a prompting editor.

### Syntax



```
prompter -help
```

AJ2FL250

### Description

The **prompter** command is used to invoke the prompter editor for message entry. **prompter** is not designed to be run directly by the user; it is designed to be called by other programs. The **prompter** command is part of the MH (Message Handling) package.

The **prompter** command opens the specified file and scans it for empty components such as **To:** and prompts you to fill in those fields. If you press **Enter** without adding text, **prompter** later deletes the component.

After the first blank line or line of dashes in the file, **prompter** accepts text for the body of the message. If the body already contains text and the flag **-noprepend** is specified, **prompter** displays the text followed by the message:

```
-----Enter additional text
```

**prompter** appends any text that is entered to the message body. If you specify the **-prepend** flag, **prompter** displays the following message instead:

```
-----Enter initial text
```

When you press **END OF FILE**, **prompter** ends text entry and returns control to the calling program.

## Flags

<b>-erase</b> <i>char</i>	Sets the character to be used as the erase character. You can specify the octal representation of the character in the form <code>\nnn</code> , or you can specify the character itself.
<b>-help</b>	Displays help information for the command.
<b>-kill</b> <i>char</i>	Sets the character to be used as the kill character. You can specify the octal representation of the character in the form <code>\nnn</code> , or you can specify the character itself.
<b>-noprepend</b>	Places additional text below any text already in the message body.
<b>-norapid</b>	Displays any text already in the message body. This is the default.
<b>-prepend</b>	Places additional text before any text already in the message body. This is the default.
<b>-rapid</b>	Does not display text already in the message body.

## Profile Entries

<b>Msg-Protect:</b>	Sets the protection level for your new message files.
<b>prompter-next:</b>	Specifies the editor used after exiting <b>prompter</b> .

## Files

<code>\$HOME/.mh_profile</code>	The MH user profile.
<code>/tmp/prompter*</code>	A temporary copy of a message.

## Related Information

Other MH commands: “**comp**” on page 185, “**dist**” on page 336, “**forw**” on page 438, “**repl**” on page 821, “**whatnow**” on page 1215.

The **mh-profile** file in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

## proto

---

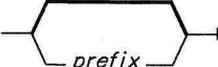
## proto

---

### Purpose

Constructs a prototype file for a file system.

### Syntax

`/etc/proto` — *directory* —  —

OL805007

### Description

The **proto** command makes a prototype file for a file system or part of a file system. Use the prototype file as input to the **mkfs** command to construct a file system according to a predefined template. The prototype file consists of a recursive directory listing of every file on the file system, with its owner, group, and protection. It also contains the file from which the prototype file is to be initialized, formatted as described in the **mkfs** command.

Specify the base directory from which the prototype file is made with *directory*. The prototype file includes the complete subtree below *directory* that is contained on the same file system as *directory*.

The *prefix* parameter is added to the names of all the initialization files, forcing the initialization files to be taken from a place other than the prototype. Before the output from **proto** can be used with **mkfs**, **mkfs** needs a start up program, a file system size, and an i-list size. Link information is not preserved with the **proto** command.

The collating sequence is determined by the `ct_collate` array in the **NLctab** subroutine.

---

#### Japanese Language Support Information

This command has not been modified to support Japanese characters.

---

### Related Information

The following command: “**mkfs**” on page 658.

“Overview of International Character Support” in *Managing the AIX Operating System*.

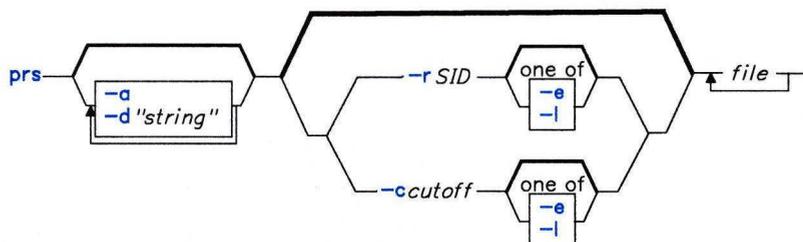
---

**prs**


---

**Purpose**

Displays a Source Code Control System (SCCS) file.

**Syntax**


OL805248

**Description**

The **prs** command reads *files*, and writes to standard output a part or all of a **Source Code Control System (SCCS)** file. If you specify a directory in place of *file*, **prs** performs the requested actions on all SCCS files (those with a name that has the **s.** prefix). If you specify a - (minus) in place of *file*, **prs** reads standard input and interprets each line as the name of an SCCS file. **prs** continues to take input until it reads an end-of-file character.

**Data Keywords**

Data keywords specify which parts of an SCCS file are to be retrieved and written to standard output. All parts of an SCCS file have an associated data keyword. There is no limit to the number of times a data keyword can appear in a *string*. The information that **prs** displays consists of user-supplied text and appropriate values (extracted from the SCCS file) substituted for the recognized data keywords in the order of appearance in *string*. The format of a data keyword value is either simple, in which the keyword substitution is direct, or multiline, in which the substitution is followed by a carriage return. Text is any characters other than recognized data keywords. Specify a tab character with `\t` and a carriage return/new-line character with a `\n`. Remember to quote the `\t` and `\n` with an extra `\` to prevent shell from interpreting the `\` and just passing a `t` or `n` to **prs** as text.

The following table lists the keywords associated with information in the delta table in the SCCS file (see the **sccsfile** file in *AIX Operating System Technical Reference* for the structure of an SCCS file).

Keyword	Data Represented	Value	Format
:R:	Release number	num	Simple
:L:	Level number	num	Simple
:B:	Branch number	num	Simple
:S:	Sequence number	num	Simple
:I:	SCCS ID string (SID)	:R::L::B::S:	Simple
:Dy:	Year delta created	YY	Simple
:Dm:	Month delta created	MM	Simple
:Dd:	Day delta created	DD	Simple
:D:	Date delta created	YY/MM/DD	Simple
:Th:	Hour delta created	HH	Simple
:Tm:	Minute delta created	MM	Simple
:Ts:	Second delta created	SS	Simple
:T:	Time delta created	HH:MM:SS	Simple
:DT:	Delta type	D or R	Simple
:P:	User who created the delta	login name	Simple
:DS:	Delta sequence number	num	Simple
:DP:	Previous delta sequence number	num	Simple
:Dt:	Delta information	:DT::I::D: :T::P::DS::DP:	Simple
:Dn:	Sequence numbers of deltas included	:DS: . . .	Simple
:Dx:	Sequence numbers of deltas excluded	:DS: . . .	Simple
:Dg:	Sequence numbers of deltas ignored	:DS: . . .	Simple
:DI:	Sequence numbers of deltas included, excluded, and ignored	:Dn:/:Dx:/:Dg:	Simple
:Li:	Lines inserted by Delta	num	Simple
:Ld:	Lines deleted by Delta	num	Simple
:Lu:	Lines unchanged by Delta	num	Simple
:DL:	Delta line statistics	:Li:/:Ld:/:Lu:	Simple

Figure 7. Delta Table Keywords

Keyword	Data Represented	Value	Format
:MR:	MR numbers for delta	text	Multiline
:C:	Comments for delta	text	Multiline

Figure 7 (Part 2 of 2). Delta Table Keywords

The following table lists the keywords associated with the header flags in the SCCS file. For more information of Header flags, see Figure 1 on page 44.

Keyword	Data Represented	Value	Format
:Y:	Module type	text	simple
:MF:	MR validation flag set	yes or no	Simple
:MP:	MR validation program name	text	Simple
:KF:	Keyword/error warning flag set	yes or no	Simple
:BF:	Branch flag set	yes or no	Simple
:J:	Joint edit flag set	yes or no	Simple
:LK:	Locked releases	:R: . . .	Simple
:Q:	User defined keyword	text	Simple
:M:	Module name	text	Simple
:FB:	Floor boundary	:R:	Simple
:CB:	Ceiling boundary	:R:	Simple
:Ds:	Default SID	:I:	Simple
:ND:	Null Delta flag set	yes or no	Simple
:FL:	Header flag list	text	Multiline

Figure 8. Header Flag Keywords

The following table lists the keywords associated with other parts of the SCCS file.

Keyword	Data Represented	Value	Format
:UN:	User names	text	Multiline
:FD:	Descriptive text	text	Multiline
:BD:	Body of text	text	Multiline
:GB:	Text in a g-file	text	Multiline
:W:	A <b>what</b> string	:Z::M: \tab :I:	Simple
:A:	A <b>what</b> string	:Z::Y::M::I::Z:	Simple
:Z:	A <b>what</b> string delimiter	@ (#)	Simple
:F:	SCCS file name	text	Simple
:PN:	SCCS file path name	text	Simple

Figure 9. Other Keywords

## Flags

Each flag or group of flags applies independently to each named file.

- a Writes information for the specified deltas, whether or not they have been removed (see “**rmdel**” on page 837). If you do not specify the **-a** flag, **prs** supplies information only for the specified deltas that have not been removed.
- ccutoff Specifies a *cutoff* date and time for the **-e** and **-l** flags. Specify *cutoff* in the following form:  
 YY[MM[DD[HH[MM[SS]]]]]  
 All omitted items default to their maximum values, so specifying **-c8402** is the same as specifying **-c840229235959**. You can separate the fields with any non-numeric characters. For example, you can specify **-c84/2/20,9:22:25** or **-c"84/2/20 9:22:25"** or **"-c84/2/20 9:22:25"**.
- d"*string*" Specifies the data items to be displayed. *string* is a string consisting of optional text and SCCS file data keywords. You must enclose all text and spaces in *string* in quotation marks.
- e Requests information for all deltas created *earlier* than and including the delta specified by the **-r** flag.

- l** Requests information for all deltas created *later* than and including the delta specified by the **-r** flag.
- rSID** Specifies the *SID* of a delta for which **prs** will retrieve information. If no *SID* is specified, **prs** retrieves the information for the *SID* of the highest numbered delta.

## Files

/tmp/pr????

## Related Information

The following commands: “**admin**” on page 41, “**delta**” on page 310, “**get**” on page 477, and “**help**” on page 513.

The **sccsfile** file in *AIX Operating System Technical Reference*.

The discussion of SCCS in *AIX Operating System Programming Tools and Interfaces*.

**ps**

---

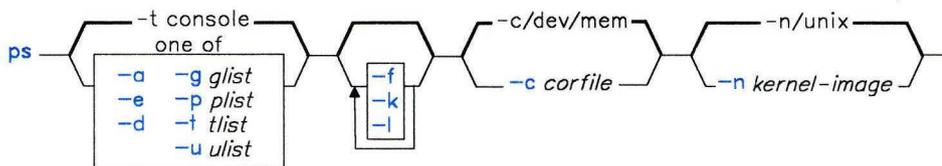
**ps**

---

## Purpose

Reports process status.

## Syntax



OL805439

## Description

The `ps` command writes certain information about active processes to standard output. Without flags, `ps` displays information about the current work station.

The column headings in a `ps` listing have the following meaning. The letters `f` and `l` following the column heads indicate which flags cause the corresponding heading to appear. If `all` follows the column head, that heading always appears. Note that the `-f` and `-l` flags determine only what information is provided about a process; they do not determine which processes are listed.

### F (l)

Flags (octal and additive) associated with the process:

- 01 In core
- 02 System process
- 04 Locked in core (for example, for physical I/O);
- 10 Waiting for a page default, or forking
- 20 Being traced by another process
- 40 Another tracing flag
- 100 Process has shared text.

### S (l)

The state of the process:

- 0 Nonexistent
- S Sleeping
- W Waiting
- R Running

---

I Intermediate  
Z Canceled  
T Stopped  
K Available kernel process  
X Growing.

**UID (f,l)**

The user ID of the process owner; the login name is displayed with the **-f** flag.

**PID (all)**

The process ID of the process.

**PPID (f,l)**

The process ID of the parent process.

**C (f,l)**

Processor utilization for scheduling.

**STIME (f)**

Starting time of the process. The **NLLDATE** and **NLTIME** environment variables control the appearance of this field.

**PRI (l)**

The priority of the process; higher numbers mean lower priority.

**NI (l)**

Nice value; used in calculating priority.

**ADDR (l)**

The segment number of the process stack, if normal; if a kernel process, the address of the preprocess data area.

**SZ (l)**

The size in blocks of the core image of the process.

**WCHAN (l)**

The event for which the process is waiting or sleeping; if blank, the process is running.

**TTY (all)**

The controlling work station for the process.

**TIME (all)**

The total execution time for the process.

**CMD (all)**

The command name; the full command name and its parameters are displayed with the **-f** flag.

A process that has exited and has a parent, but has not yet been waited for by the parent, is marked **<defunct>**.

With the **-f** flag, **ps** determines what the command name and parameters were when the process was created by examining memory or the paging area. If it cannot find this information, the command name, as it would appear without the **-f** flag, displays in square brackets.

**Notes:**

1. The process can change while **ps** is running.
2. Some data displayed for defunct processes are irrelevant.
3. The current work station is defined as the one associated with standard error. Thus redirecting standard error, for example:

```
ps 2> /dev/null
```

does not work as expected.

**Flags**

- |                               |  |
|-------------------------------|--|
| <b>-a</b>                     | Writes to standard output information about all processes except the process group leaders and processes not associated with a terminal.   |
| <b>-c</b> <i>corefile</i>     | Uses <i>corefile</i> instead of the default <b>/dev/mem</b> . <i>corefile</i> is a core image file that has been created by the <b>Ctrl-(left)Alt-End</b> key sequence.  |
| <b>-d</b>                     | Writes information to standard output about all processes except the process group leaders.  |
| <b>-e</b>                     | Writes information to standard output about all processes except kernel processes.   |
| <b>-f</b>                     | Generates a full listing. The meaning of columns in a full listing is described on page 786.   |
| <b>-g</b> <i>glist</i>        | Writes information to standard output only about processes that are in the process groups listed in <i>glist</i> . The <i>glist</i> is either a comma-separated list of process-group identifiers or a list of process-group identifiers enclosed in double quotation marks (" ") and separated from one another by a comma and/or one or more spaces. |
| <b>-k</b>                     | Writes information to standard output about kernel processes. Otherwise, it does not list kernel processes.  |
| <b>-l</b>                     | Generates a long listing. The meaning of a long listing is described on page 786.  |
| <b>-n</b> <i>kernel-image</i> | Takes <i>kernel-image</i> as the name of an alternate <i>kernel-image</i> file ( <b>/unix</b> is the default).   |

- 
- p** *plist* Displays only information about processes with the process numbers specified in *plist*. *plist* is either a comma-separated list of process-ID numbers or a list of process-ID numbers enclosed in double quotation marks (" ") and separated from one another by a comma and/or one or more spaces.
- t** *tlist* Displays only information about processes associated with the work stations listed in *tlist*. *tlist* is either a list of comma-separated work-station identifiers or a list of work-station identifiers enclosed in double quotation marks (" ") and separated from one another by a comma and/or one or more spaces.
- u** *ulist* Displays only information about processes with the user ID numbers or login names specified in *ulist*. *ulist* is either a comma-separated list of user ID's or a list of user ID's enclosed in double quotation marks (" ") and separated from one another by a comma and/or one or more spaces. In the listing, **ps** displays the numerical user ID unless the **-f** flag is used; then it displays the login name.

## Examples

1. To list the processes that you have started:

```
ps
```

This command displays a summary of information about the processes associated with your work station.

2. To display all process information available:

```
ps -e -f -l
```

This command displays all of the information (-l -f) about all processes (-e).

3. To list processes owned by specific users:

```
ps -f -l -ujim,jane,su
```

This command displays all the information available (-l -f) about the processes being run by the users jim, jane, and su.

4. To list processes associated with specific work stations:

```
ps -t-,console
```

This command displays information about processes not connected to any work station (-t-), and processes associated with the work station /dev/console.

## Files

/unix	System kernel image.
/dev/mem	Memory.
/etc/passwd	Supplies UID information.
/etc/ps_data	Internal data structure.
/dev	Searched to find work station ( <b>tty</b> ) names.

## Related Information

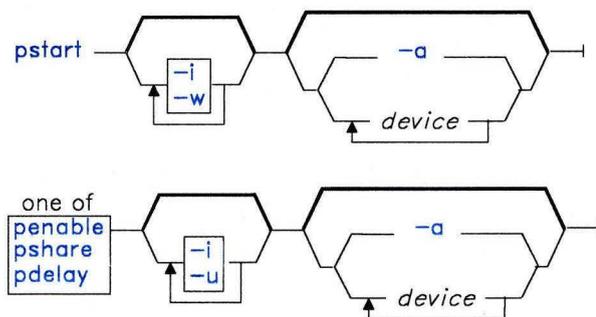
The following commands: “**kill**” on page 552 and “**nice**” on page 699.

## pstart, penable, pshare, pdelay

### Purpose

Enables or reports the availability of login ports.

### Syntax



OL805208

### Description

The `pstart`, `penable`, `pshare`, and `pdelay` commands each enable a set of login ports in the `/etc/ports` file. Enabling a port makes the port available to log in. The system enables a port by updating an entry in the `/etc/portstatus` file and then sending a signal to `init`. When `init` receives the signal and reads the updated status entry, it takes the appropriate action.

Use the `device` parameter to specify the ports to be enabled. Permitted values for `device` include:

- A full device name, such as `/dev/tty1`.
- A simple device name, such as `tty1`.
- A general class of devices in the form `attribute=value`, which is equivalent to naming each port with a stanza in `/etc/ports` that includes the specified attribute).

If you do not specify a `device` to enable, each command reports the names of currently enabled ports in its set.

### pstart

The **pstart** command enables all ports (normal, shared, and delayed) that are enabled in the `/etc/ports` file. If you do not specify a *device* to enable, **pstart** reports the names of all enabled ports and tells whether they are currently enabled as normal, shared, or delayed. Usually the command is run in the form **pstart -a -i -w** from `/etc/rc` to enable all ports on a multiuser system.

### penable

The **penable** command enables normal ports that are enabled in the `/etc/ports` file. Normal ports are ports that are asynchronous and only allow users to login to those ports. No outgoing use of the port is allowed while it is enabled. This command is equivalent to the statement **penable enabled = true**. If you do not specify a *device*, **penable** reports the names of the currently enabled normal ports.

### pshare

The **pshare** command enables shared ports that are enabled in the `/etc/ports` file. Shared ports are bidirectional. This command is equivalent to the statement **pshare enabled = share**. If you do not specify a *device*, **pshare** reports the names of the currently enabled shared ports. To enable shared ports, **getty** attempts to create a lock file in `/etc/locks` which contains the ASCII process ID of the **getty** process. If the port is already in use by some other process, **getty** waits until the port is available and tries again.

### pdelay

The **pdelay** command enables delayed ports that are enabled in the `/etc/ports` file. Delayed ports are ports that are enabled like shared ports except that the login herald is not displayed until the user types one or more characters (usually carriage returns). If the port is directly connected to a remote system or connected to an intelligent modem, the port is usually enabled as a delayed port to prevent the **getty** from talking to a **getty** on the remote side or to the modem on a local connection, thereby consuming system resources. This statement is equivalent to **pdelay enabled = delay**. If you do not specify a *device*, **pdelay** reports the names of the currently enabled delayed ports.

## Flags

- a With **pstart**, this flag enables all ports enabled in the `/etc/ports` file (normal, shared, and delayed ports). With **penable**, this flag enables all normal ports that are enabled in the `/etc/ports` file. With **pshare**, this flag enables all shared ports that are enabled in the `/etc/ports` file. With **pdelay**, this flag enables all delayed ports that are enabled in the `/etc/ports` file.
- i Reinitializes an existing `/etc/portstatus` file instead of updating the existing one. You typically use this flag in the `/etc/rc` command file to re-establish default port enabling before starting up the system with multiple users.

- w Makes the command return immediately rather than wait for **init** to confirm the changes in port status. You must use this flag when running **pstart**, **penable**, **pshare**, or **pdelay** either in maintenance mode or from **/etc/rc** because **init** does not initiate loggers until the system is in normal mode.

## Examples

1. To display the names of all ports (normal, shared, and delayed) currently enabled and how they are enabled:  
`pstart`
2. To enable all normal, shared, and delayed ports that are enabled in **/etc/ports**, reinitialize the existing **/etc/ports**, and make the command return immediately rather than wait for **init** to confirm port status:  
`pstart -a -i -w`
3. To enable the work station attached to the **/dev/tty2** port as a shared port:  
`pshare /dev/tty2`
4. To display the names of the delayed ports that are currently enabled:  
`pdelay`
5. To enable the work station attached to the **/dev/tty8** port as a shared port and return immediately without waiting to confirm the changes in the port status:  
`pshare -w tty8`
6. To enable all 9600 baud ports as delayed:  
`pdelay speed=9600`

## Files

<code>/etc/locks</code>	Contains lock files for <b>pshare</b> and <b>pdelay</b> .
<code>/etc/ports</code>	Contains descriptions of known normal, shared, and delayed ports.
<code>/etc/portstatus</code>	Contains current status of each known login port.

## Related Information

The following commands: “**init**” on page 521 and “**pdisable**, **phold**” on page 741.  
The **ports** and **portstatus** files in *AIX Operating System Technical Reference*.

# ptx

---

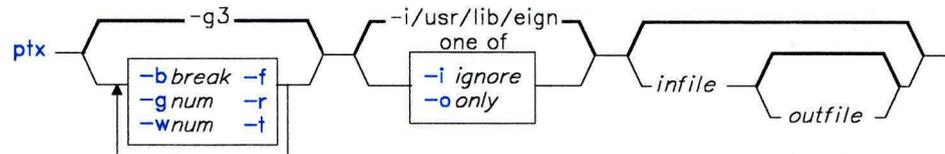
## ptx

---

### Purpose

Generates a permuted index.

### Syntax



OL805250

### Description

The **ptx** command reads *infile* (standard input by default), creates a permuted index from its input, and writes to *outfile* (standard output by default).

The **ptx** command searches *infile* for keywords, sorts the lines, and generates the file *outfile*. *outfile* can then be processed with **nroff** or **troff** to produce a permuted index from the file *infile*.

The **ptx** command follows three steps:

1. In the permutation, generates one line for each keyword in an input line, and rotates the keyword to the front.
2. Sorts the permuted file.
3. Rotates the sorted lines so that the keyword comes at the middle of each line.

The resulting lines in output are in the form:

`.XX, "tail" "before keyword" "keyword and after" "head"`

where `.XX` is an **nroff** or **troff** macro provided by the user, or provided by the **mptx** macro package (see the *AIX Operating System Technical Reference* for information on this macro package). The *before keyword* and *keyword and after* fields incorporate as much of the line as will fit around the keyword when it is printed. *tail* or *head*, at least one of which is always the empty string, are wrapped-around pieces small enough to fit in the unused space at the opposite end of the line.

**Notes:**

1. Line length counts do not account for overstriking or proportional spacing.
2. Lines that contain tildes (~) do not work because **ptx** uses that character internally.

**Flags**

- b *break* Uses the characters in the *break* file to separate words. Tab characters, new-line characters, and spaces are **always** used as break characters.
- f Does not distinguish between uppercase and lowercase characters while sorting (see “**sort**” on page 958).
- g *num* Uses *num* as the number of spaces displayed between the four parts of the line. The default *num* is 3.
- i *ignore* Does not use any words in the *ignore* file as keywords. If the -i and -o flags are not used, **/usr/lib/eign** is the default *ignore* file.
- o *only* Uses only the words in the *only* file as keywords.
- r Takes any leading nonblank characters of each input line to be a reference identifier separate from the text of the line. Attaches that identifier as a fifth field on each output line.
- t Prepares the output for the phototypesetter.
- w *num* Uses *num* as the length of the output line. The default line length is 72 characters for **nroff** and 100 for **troff**.

**Files**

/bin/sort  
/usr/lib/eign  
/usr/lib/tmac/tmac.ptx

**Related Information**

The following commands: “**nroff**, **troff**” on page 709 and “**troff**” on page 710.

The **mm** and **mptx** miscellaneous facilities in *AIX Operating System Technical Reference*.

“Overview of International Character Support” in *Managing the AIX Operating System*.

## puttext

---

## puttext

---

### Purpose

Updates an output file that contains message/insert/help descriptions.

### Syntax

```
puttext -n infile outfile
```

OL805209

### Description

The **puttext** command uses the message/insert/help descriptions in *infile* to change, delete and add message/insert/help text to *outfile* for a component. (For information about the format and contents of *infile*, see *AIX Operating System Programming Tools and Interfaces*.)

The *infile* parameter specifies the name of the file where the message/insert/help descriptions reside. See *AIX Operating System Programming Tools and Interfaces* for a discussion of the **gettext** output file parameters that describes the format and contents of this file.

The *outfile* parameter specifies the name of the output file. If you specify an *outfile* that does not exist, a new component file is created. If you specify an existing *outfile*, a copy of that file is renamed as a backup file. In this case, an old backup file will be deleted.

**Note:** In order for the new file to be accessed by the message support run-time services, the output file name must be in the format *xxxccc-EN.m*. If you do not specify *outfile*, the component ID is prefixed to *-EN.m* to form the output file name.

### Flag

- n** Causes **puttext** to assign available index numbers to the input descriptions. If you specify this flag, all the index number fields of the input file must be underscore characters or blanks.

## **Related Information**

The following commands: “**gettext**” on page 488.

The discussion of **puttext** in *AIX Operating System Programming Tools and Interfaces*.

# pwck

---

# pwck

---

## Purpose

Checks the password and group files for inconsistencies.

## Syntax

`pwck` `/etc/passwd`  
`file`

OL805008

`grpck` `/etc/group`  
`file`

OL805558

## Description

The **pwck** command scans the named *file* or the default file **/etc/passwd** and writes to standard output any inconsistencies. The checks include validation of the number of fields, login name, user ID, group ID, and existence of the login directory and optional program name.

The **grpck** command scans the named *file* or the default file **/etc/group** and writes to standard output any inconsistencies. The checks include validation of the number of fields, group name, group ID, and whether all login names appear in the password file. The **grpck** command writes to standard output any group entries that do not have login names.

---

### Japanese Language Support Information

This command has not been modified to support Japanese characters.

---

## Files

<code>/etc/passwd</code>	Password file; contains user IDs.
<code>/etc/group</code>	Group file; contains group IDs.

## Related Information

The discussion of passwords in *Managing the AIX Operating System*.

## **pwd**

---

## **pwd**

---

### **Purpose**

Displays the path name of the working directory.

### **Syntax**

`pwd` —

OL805210

### **Description**

The **pwd** command writes to standard output the full path name of your current directory (from the root directory). All directories are separated by a / (slash). The root directory is represented by the first /, and the last directory named is your current directory.

### **Related Information**

The following command: “**cd**” on page 150.

The **fullstat** and **ffullstat** system calls in *AIX Operating System Technical Reference*.

# **pwttable**

---

## **Purpose**

Accesses the Distributed Services Node Security Table.

## **Syntax**

`pwttable` —|

AJ2FL144

## **Description**

The **pwttable** command lets you build, examine, or change the Distributed Services Network Node Security Table. Only members of the system group or users operating with superuser authority can use **pwttable** to change the state of the table (see “**su**” on page 1026). Other users can use **pwttable** to browse through the table.

## **Related Information**

“Getting Started With Distributed Services Configuration Menus” in *Managing the AIX Operating System*.

# qdaemon

---

## qdaemon

---

### Purpose

Schedules jobs enqueued by the **print** command.

### Syntax

`qdaemon` <sup>1</sup>

<sup>1</sup> This command is not usually entered on the command line.

OL805148

### Description

The **qdaemon** is a background process (usually started by the **rc** command file) that schedules printing jobs enqueued by **print**.

### Files

<code>/usr/lpd/qdir/*</code>	Print requests.
<code>/usr/lpd/stat/*</code>	Information on the status of the devices.
<code>/usr/spool/qdaemon/*</code>	Temporary copies of files to be printed.

### Related Information

The following commands: “**lp**” on page 593, “**piobe**” on page 753, and “**print**” on page 767.

The **qconfig** file in *AIX Operating System Technical Reference*.

---

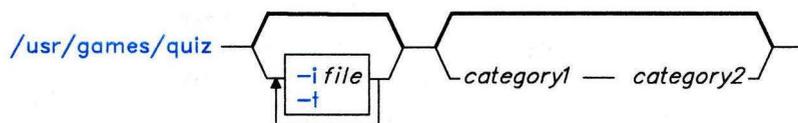
## quiz

---

### Purpose

Tests your knowledge.

### Syntax



OL805230

### Description

The **quiz** game gives associative knowledge tests on various selectable subjects. It asks about items chosen from *category1* and expects answers from *category2*. If you do not specify the categories, **quiz** gives instructions and lists the available categories.

The **quiz** game gives the correct answer whenever you press the **Enter** key by itself. The game ends when questions run out or when you press INTERRUPT (**Alt-Pause**); **quiz** reports a score and exits.

### Flags

**-i***file* Substitutes the named *file* for the standard index file.

**Note:** In the following syntax description, brackets are normally used to indicate that an item is optional; a bold-faced bracket or brace, however, should be entered as a literal part of the syntax. A vertical list of items indicates that one and only one must be chosen. The lines in *file* must have the following syntax:

```
line      = category [ :category ] . . .
category  = alternate [ |alternate ] . . .
alternate = [primary]
primary   = character
           [category]
           option
option    = {category}
```

## quiz

---

In an index file, the first category of each line must specify the name of an information file (the information file contains the names of files with quiz material). The remaining categories specify the order and contents of the data in each line of the information file. The quiz data in an information file follows the same syntax. A \ (backslash) is an escape character which allows you to quote syntactically significant characters or to insert a new-line character (\n) into a line. When either a question or its answer is blank, **quiz** does not ask it. The construct **a|ab** does not work in an information file. Use **a{b}**.

- t Provides a tutorial. Repeats missed questions and introduce new material gradually.

## Examples

1. To start a Latin-to-English quiz:

```
/usr/games/quiz latin english
```

The **quiz** command displays Latin words and waits for you to enter what they mean in English.

2. To start an English-to-Latin quiz:

```
/usr/games/quiz english latin
```

3. To set up a Latin-English quiz, add the following line to the index file:

```
/usr/games/lib/quiz/latin:latin:english
```

This line specifies that the file `/usr/games/lib/quiz/latin` contains information about the categories `latin` and `english`.

You can add new categories to the standard index file, `/usr/games/lib/quiz/index`, or to an index file of your own. If you create your own index file, run the **quiz** command with the **-i file** flag to give it your list of quiz topics.

4. This is a sample information file:

```
cor:heart
sacerdos:priest{ess}
quando:when|since|because
optat:{{s}he |it |[desires|wishes]\
|desire|wish
alb[ustium]:white
```

This information file contains Latin and English words. The `:` (colon) separates each Latin word from its English equivalent. Items enclosed in `{ }` (braces) are optional. A `|` (vertical bar) separates two items when entering either is correct. The `[ ]` (brackets) group items separated by vertical bars.

---

The first line accepts only the answer `heart` in response to the Latin word `cor`. The second accepts either `priest` or `priestess` in response to `sacerdos`. The third line accepts `when`, `since`, or `because` for `quando`.

The `\` (backslash) at the end of the fourth line indicates that this entry continues on the next line. In other words, the fourth and fifth lines together form one entry. This entry accepts any of the following in response to `optat`:

```
she desires    it desires    desire
she wishes    it wishes    wish
he desires     desires
he wishes     wishes
```

If you start a Latin-to-English quiz, then the last line of this sample information file instructs **quiz** to ask you the meaning of `albus`. If you start an English-to-Latin quiz, then **quiz** displays `white` and accepts `albus`, `alba`, or `album` for the answer.

If any of the characters `{`, `}`, `[`, `]`, or `|` appear in a question item, then **quiz** gives the first alternative of every `|` group and displays every optional group. Thus, the English-to-Latin question for the fourth definition in this sample is `she desires`.

## Files

```
/usr/games/lib/quiz/index
/usr/games/lib/quiz/*
```

**rc**

---

**rc**

---

## Purpose

Performs normal startup initialization.

## Syntax

`/etc/rc`  $\rightarrow$  <sup>1</sup>

---

<sup>1</sup> This command is not usually run from the command line.

OL805339

## Description

When the **init** process starts up the system in normal operating mode, it runs the command file `/etc/rc` to perform the necessary system initialization, including the enabling of various loggers. If the system is being brought up with no file system checking, **init** passes the argument **m** to **rc**. If **init** determines that the root file system needs consistency checking, it passes the argument **d** to **rc**.

The contents of `/etc/rc` may be installation specific, but there are a few things that it should do:

- Check the default file systems if **rc** is passed the **init -d** flag (Run **fsck**)
- Mount the default file systems (Run **mount**)
- Purge temporary files
- Start SNA and Distributed Services (Run `/etc/rc.sna` and `/etc/rc.ds`)
- Set printer defaults
- Enable default ports (Run **pstart**)
- Determine whether to set up stand-alone or active-service system (Run **chngstate**)
- Start the error daemon and run `/etc/rc.include`.

If all of the necessary operations complete successfully, the file exits with a zero return code that allows **init** to start loggers to complete normal initialization and startup.

---

**Notes:**

1. Many system daemons such as **cron** are started by **rc** indirectly when it runs **/etc/rc.include**.
2. The mail facility is started by **rc** indirectly when it runs **/etc/rc.include** and **/etc/rc.tcPIP**.
3. The root file system is implicitly mounted.

**Files**

<code>/etc/rc.ds</code>	Performs functions required to start Distributed Services.
<code>/etc/rc.include</code>	Performs functions required to start most program daemons.
<code>/etc/rc.sna</code>	Performs functions required to start SNA.
<code>/etc/rc.tcPIP</code>	Performs functions required to start TCPIP.

**Related Information**

The following commands: “**chgstate**” on page 164, “**cron**” on page 220, “**fsck, dfscck**” on page 445, “**init**” on page 521, “**mount**” on page 669, and “**pstart, penable, pshare, pdelay**” on page 791.

The discussion of starting up the system in *Managing the AIX Operating System*.

# rcvdist

---

## rcvdist

---

### Purpose

Sends a copy of incoming messages to additional recipients.

### Syntax

```
/usr/lib/mh/rcvdist user
```

```
/usr/lib/mh/rcvdist -help
```

AJ2FL234

### Description

The **rcvdist** command is used to forward copies of incoming messages to other users. **rcvdist** is not designed to be run directly by the user; it is designed to be called by `/usr/lib/mh/slocal`. The **rcvdist** command is part of the MH (Message Handling) package.

The **rcvdist** command sends a copy of the incoming message to the specified users. **rcvdist** uses the format string facility described in **mh-format**. You can run **rcvdist** on all incoming messages by specifying the **rcvdist** command in the `.maildelivery` file.

### Flag

**-help** Displays help information for the command.

### Files

<code>\$HOME/.maildelivery</code>	The user's local mail delivery instructions.
<code>\$HOME/.forward</code>	The user's default message filter.

## Related Information

Other commands: “**ali**” on page 48, “**rcvpack**” on page 810, “**rcvstore**” on page 812, “**rcvtty**” on page 815, “**sendmail**” on page 897, “**slocal**” on page 954, and “**whom**” on page 1222.

The **mh-alias**, **mh-format**, **mh-mail**, and **mh-profile** files in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

# rcvpack

---

## rcvpack

---

### Purpose

Saves incoming messages in a packed file.

### Syntax

```
rcvpack file
```

```
rcvpack --help
```

AJ2FL235

### Description

The **rcvpack** command is used to place incoming messages in a packed file. **rcvpack** is not designed to be run directly by the user; it is designed to be called by **/usr/lib/mh/slocal**. The **rcvpack** command is part of the MH (Message Handling) package.

The **rcvpack** command appends a copy of the incoming message to the specified file and runs the **packf** command on the file. You can run **rcvpack** on all incoming messages by specifying the **rcvpack** command in the **.maildelivery** file.

### Flag

**-help** Displays help information for the command.

### Files

<b>\$HOME/.maildelivery</b>	The user's local mail delivery instructions.
<b>\$HOME/.forward</b>	The user's default message filter.

## Related Information

The following commands: “**inc**” on page 518, “**packf**” on page 733, “**rcvdist**” on page 808, “**rcvstore**” on page 812, “**rcvtty**” on page 815, “**sendmail**” on page 897, and “**slocal**” on page 954.

The **mh-alias**, **mh-format**, **mh-mail**, and **mh-profile** files in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

## rcvstore

---

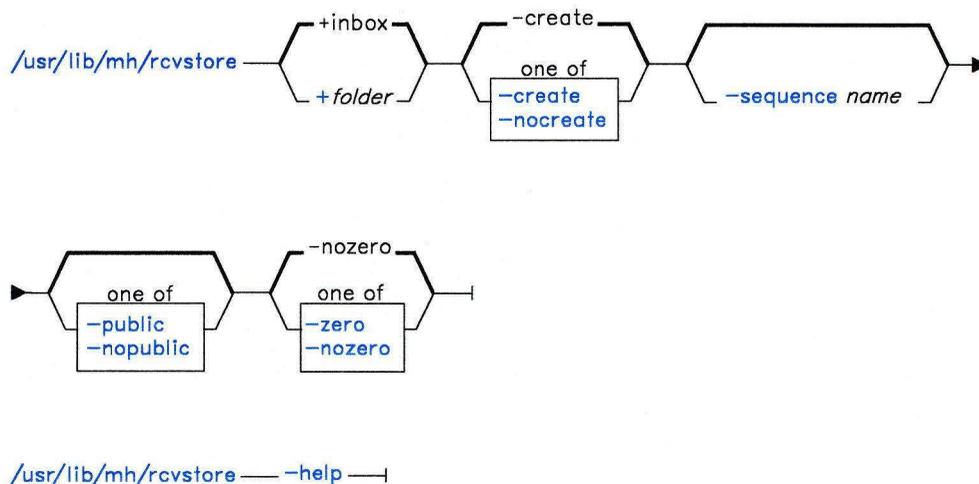
## rcvstore

---

### Purpose

Incorporates new mail from standard input into a folder.

### Syntax



AJ2FL237

### Description

The `rcvstore` command is used to incorporate incoming messages. `rcvstore` is not designed to be run directly by the user; it is designed to be called by `/usr/lib/mh/slocal`. The `rcvstore` command is part of the MH (Message Handling) package.

The `rcvstore` command accepts messages from standard input and places them in a specified folder. You can run `rcvstore` on all incoming messages by specifying the `rcvstore` command in the `.maildelivery` file.

You can specify `rcvstore` flags in `$HOME/.maildelivery` or as with most MH commands, in `$HOME/.mh-profile`.

## Flags

<b>-create</b>	Creates the specified folder in your mail directory if the folder does not exist.
<b>+folder</b>	Places the incorporated messages in the specified folder. The default is <b>+inbox</b> .
<b>-help</b>	Displays help information for the command.
<b>-nocreate</b>	Does not create the specified folder if the folder does not exist.
<b>-npublic</b>	Restricts the specified sequence to your usage. <b>-npublic</b> does not restrict the messages in the sequence, only the sequence. This flag is the default if the folder is write-protected from other users.
<b>-nozero</b>	Appends the messages incorporated by rcvstore to the specified sequence (see the <b>-zero</b> flag).
<b>-public</b>	Makes the specified sequence available to other users. <b>-public</b> does not make protected messages available, only the sequence. This flag is the default if the folder is not write-protected from other users.
<b>-sequence name</b>	Adds the incorporated messages to the specified sequence.
<b>-zero</b>	Clears the specified sequence before placing the incorporated messages into the sequence. This flag is the default (see the <b>-nozero</b> flag).

## Profile Entries

<b>Folder-Protect:</b>	Sets the protection level for your new folder directories.
<b>Msg-Protect:</b>	Sets the protection level for your new message files.
<b>Path:</b>	Specifies your <i>user-mh-directory</i> .
<b>Unseen-Sequence:</b>	Specifies the sequences used to keep track of your unseen messages.
<b>Rcvstore:</b>	Specifies flags for the rcvstore program.

## Files

<code>\$HOME/.mh-profile</code>	The MH user profile.
<code>\$HOME/.maildelivery</code>	The user's local mail delivery instructions.
<code>\$HOME/.forward</code>	The user's default message filter.

### Related Information

The following commands: “**inc**” on page 518, “**rcvdist**” on page 808, “**rcvpack**” on page 810, “**rcvtty**” on page 815, “**sendmail**” on page 897, and “**slocal**” on page 954.

The **mh-alias**, **mh-format**, **mh-mail**, and **mh-profile** files in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

---

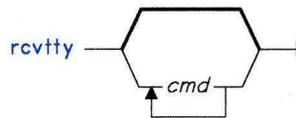
# rcvttty

---

## Purpose

Notifies the user of incoming messages.

## Syntax



`rcvttty -help`

AJ2FL238

## Description

The `rcvttty` command is used to send the user a message when incoming mail has arrived. `rcvttty` is not designed to be run directly by the user; it is designed to be called by `/usr/lib/mh/slocal`. The `rcvttty` command is part of the MH (Message Handling) package.

The `rcvttty` command sends a one-line scan listing to your terminal. If you give `rcvttty` a command as an argument, `rcvttty` executes the command with the incoming message as the command's standard input, and sends the output to the terminal. For `rcvttty` to write output to your terminal, your terminal's write permission must be set to "All".

You can run `rcvttty` on all incoming messages by specifying the `rcvttty` command in the `.mailedelivery` file.

## Flag

`-help` Displays help information for the command.

## Files

<code>\$HOME/.mailedelivery</code>	The user's local mail delivery instructions.
<code>\$HOME/.forward</code>	The user's default message filter.

### Related Information

The following commands: “**rcvdist**” on page 808, “**rcvpack**” on page 810, “**rcvstore**” on page 812, “**rcvttty**” on page 815, “**sendmail**” on page 897, and “**slocal**” on page 954.

The **mh-alias**, **mh-format**, **mh-mail**, and **mh-profile** files in *AIX Operating System Technical Reference*.

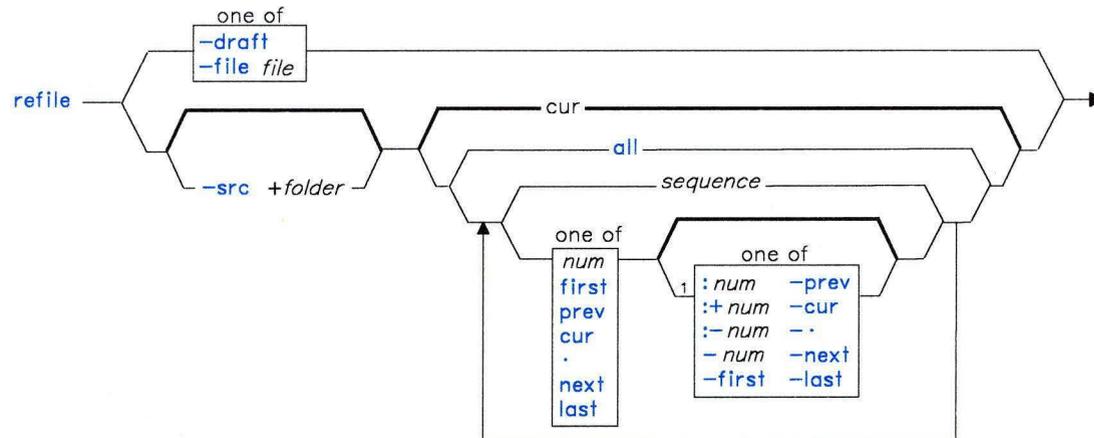
“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

# refile

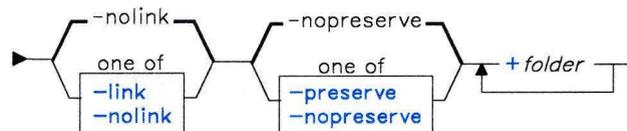
## Purpose

Files messages in other folders.

## Syntax



AJ2FL223



refile --help

AJ2FL169

<sup>1</sup> Do not put a blank between these items.

OL805308

# refile

---

## Description

The **refile** command is used to copy and move messages to other files. The **refile** command is part of the MH (Message Handling) package and can be used with MH and AIX commands.

The **refile** command copies messages or moves messages from one folder to another folder. If a destination folder does not exist, **refile** asks if it should create the folder.

## Flags

- draft** Copies the current draft message from your mail directory.
- file *file*** Copies the specified file. The file must be in valid message format. (Use the **inc** command to incorporate new messages and to format them correctly.)
- + *folder*** Copies the messages to the specified folder. Any number of folders can be specified.
- help** Displays help information for the command.
- link** Leaves the messages in the source folder or file after they are copied.
- nolink** Removes the messages from the source folder or file after they are copied. This flag is the default.
- nopreserve** Renumbers the messages that are copied. Renumbering begins with the number that is one higher than the last message in the destination folder. This is the default.
- preserve** Preserves the message numbers of the messages that are copied. If messages with those numbers already exist, **refile** issues an error message and does not alter the contents of the folders.
- src + *folder msgs*** Specifies the messages to be copied. You can use the following message references when specifying *msgs*:

<i>num</i>	<b>first</b>	<b>prev</b>
<b>cur</b>	.	<b>next</b>
<b>last</b>	<b>all</b>	<i>sequence</i>

The default message is the current message in the current folder. If a folder is specified, it becomes the current folder. If the **-link** flag and **all** are used, the current message does not change. Otherwise, if a message is specified, that message becomes the current message.

## Profile Entries

<b>Current-Folder:</b>	Sets your default current folder.
<b>Folder-Protect:</b>	Sets the protection level for your new folder directories.
<b>Path:</b>	Specifies your <i>user-mh-directory</i> .
<b>rmmproc:</b>	Specifies the program used to remove messages from a folder.

## Files

`$HOME/.mh-profile`      The MH user profile.

## Related Information

Other MH commands: “**folder**” on page 429, “**folders**” on page 433.

The **mh-profile** file in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

## regcmp

---

## regcmp

---

### Purpose

Compiles patterns.

### Syntax



OL805211

### Description

The **regcmp** command compiles the pattern in *file*, placing its output in *file.i*.

In most cases, **regcmp** makes unnecessary the use of the **regcmp** system call in your C programs, saving execution time and program size. The output of **regcmp** is C source code. Make each file entry a C variable name, followed by one or more blanks, followed by a pattern enclosed in double quotation marks (" "). Compiled patterns are initialized **char** declarations. Thus, *file.i* can be included in C programs, and *file.c* can be a file parameter to the **cc** command. The C program that uses **regcmp** output should use the **regex** subroutine to apply it to a string. (See **regcmp** and **regex** in *AIX Operating System Technical Reference*.)

### Flag

- Places the output in *file.c*

### Related Information

The **regcmp** subroutine in *AIX Operating System Technical Reference*.

---

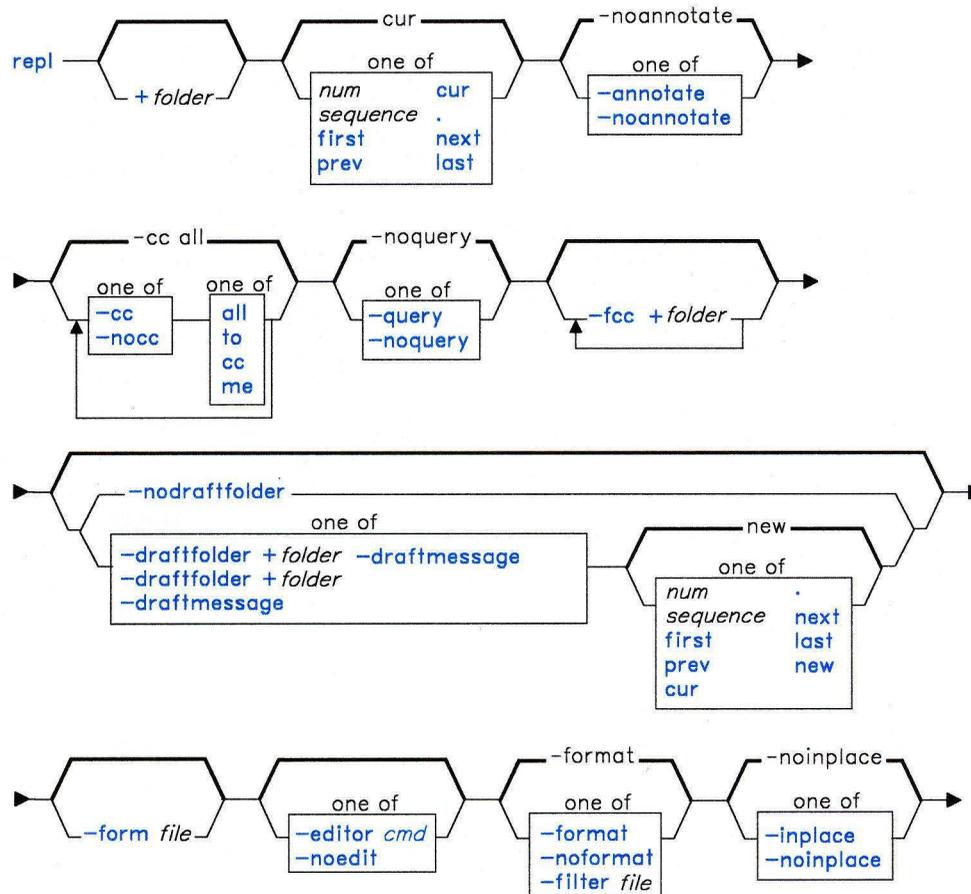
# repl

---

## Purpose

Replies to a message.

## Syntax



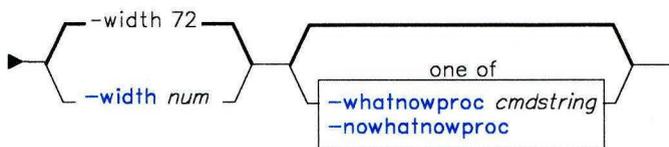
AJ2FL240

AJ2FL157

AJ2FL241

# repl

---



repl — -help —

AJ2FL226

## Description

The **repl** command is used to compose a reply to a message. **repl** is part of the MH (Message Handling) package and can be used with other MH and AIX commands.

By default, **repl** copies a message form to a new draft message and invokes an editor. You can then fill in the message header fields **To:** and **Subject:**, fill in or delete the other header fields (such as **cc:** and **Bcc:**), and add the body of the message. When you exit the editor, the **repl** command invokes the MH command **whatnow**. You can press **Enter** to see a list of the available **whatnow** subcommands. These subcommands enable you to continue editing the reply, list the reply, direct the disposition of the reply, or end the processing of the **repl** command. “**whatnow**” on page 1215 describes the subcommands.

You can specify the message that you want to reply to by using the **+folder msg** flag. If you do not specify a message, **repl** replies to the current message.

You can specify the format of the reply by using the **-form** flag. If you do not specify this flag, **repl** uses your default message format located in the file *user\_mh\_directory/replcomps*. If this file does not exist, **repl** uses the system default message format located in */usr/lib/mh/replcomps*.

**Note:** The line of dashes or a blank line must be left between the header and the body of the message for the message to be identified when it is sent.

## Flags

**-annotate**

Annotates the message being replied to with the lines:

Replied: *date*  
Replied: *addrs*

The annotation appears in the original draft message so that you can maintain a complete list of activities associated with the original message.

If you do not actually send the reply using the immediate **repl** command, the **-annotate** flag may fail to provide annotation. The **-inplace** flag forces annotation to be done in place.

- cc names** Specifies the users who will be listed in the **cc:** field of the reply. You can specify the following for *names*: **all**, **to**, **cc**, and **me**. The default is **-cc all**.
- draftfolder +folder** Places the draft message in the specified folder. If you do not specify this flag, **repl** selects a default draft folder according to the information supplied in the MH profiles. You can define a default draft folder in **\$HOME/.mh-profile**. If **-draftfolder +folder** is followed by *msg*, *msg* represents the **-draftmessage** attribute.
- draftmessage msg** Specifies the draft message. You can use one of the following message references as *msg*:

<i>num</i>	<i>sequence</i>	<b>first</b>
<b>prev</b>	<b>cur</b>	<b>.</b>
<b>next</b>	<b>last</b>	<b>new</b>

The default draft message is **new**.

- editor cmd** Specifies that *cmd* is the initial editor for composing the reply. If you do not specify this flag, **repl** selects a default editor or suppresses the initial edit, according to the information supplied in the MH profiles. You can define a default initial editor in **\$HOME/.mh-profile**.
- fcc +folder** Places a file copy of the reply in *folder*. If you do not specify this flag, **repl** will not produce a file copy.
- filter file** Reformats the message being replied to and places the reformatted message in the body of the reply. **-filter** uses the **mhl** command and the specified format file. If you do not specify this flag, **repl** will omit the original message from the reply. The **repl** command does not have a default filter file. Thus, if you specify **-filter**, you must also specify *file*.
- +folder msg** Replies to the specified message in the specified folder. You can use one of the following message references as *msg*:

<i>num</i>	<i>sequence</i>	<b>first</b>
<b>prev</b>	<b>cur</b>	<b>.</b>
<b>next</b>	<b>last</b>	

The default message is the current message in the current folder. If you specify a folder, that folder becomes the current folder. The message being replied to becomes the current message.

## repl

---

- form *file*** Uses the form contained in the specified file for the form of the reply. **repl** treats each line in *file* as a format string.
- format** Removes duplicate addresses from the fields **To:**, **cc:**, and **Bcc:** and standardizes these fields. **repl** also uses the columns specified by the **-width** flag to determine the format of these fields. This flag is the default.
- help** Displays help information for the command.
- inplace** Forces annotation to be done in place in order to preserve links to the annotated message.
- noannotate** Does not annotate the message. This flag is the default.
- nocc *names*** Specifies the users who will not be listed in the **cc:** field of the reply. You can specify the following for *names*: **all**, **to**, **cc**, and **me**.
- nodraftfolder** Places the draft in the file *user-mh-directory/draft*.
- noedit** Suppresses the initial edit.
- noformat** Does not remove duplicate addresses from the fields **To:**, **cc:**, and **Bcc:** or standardize these fields. **repl** also does not use the columns specified by the **-width** flag to determine the format of these fields.
- notinplace** Does not perform annotation in place. This flag is the default.
- noquery** Automatically builds the **To:** and **cc:** fields. This flag is the default.
- nowhatnowproc** Does not invoke a program that guides you through the reply tasks. The **-nowhatnowproc** flag also prevents any edit from occurring.
- query** Builds the **To** and **cc:** fields by interactively asking you if you want each address that would normally be placed in these fields actually placed in these fields.
- whatnowproc *cmdstring*** Invokes *cmdstring* as the program to guide you through the reply tasks. See “**whatnow**” on page 1215 for information about the default **whatnow** program and its subcommands.
- Note:** If you specify **whatnow** for *cmdstring*, **repl** invokes an internal **whatnow** procedure rather than a program with the file name **whatnow**.
- width *num*** Sets the width of the address fields. The default is 72 columns.

## Profile Entries

<b>Alternate-Mailboxes:</b>	Specifies your mailboxes.
<b>Current-Folder:</b>	Sets your default current folder.
<b>Draft-Folder:</b>	Sets your default folder for drafts.
<b>Editor:</b>	Sets your default initial editor.
<b>fileproc:</b>	Specifies the program used to refile messages.
<b>mhproc:</b>	Specifies the program used to filter the message for which you are creating a reply.
<b>Msg-Protect:</b>	Sets the protection level for your new message files.
<b>Path:</b>	Specifies your <i>user-mh-directory</i> .
<b>whatnowproc:</b>	Specifies the program used to prompt What now? questions.

## Files

<code>/usr/lib/mh/replcomps</code>	The MH default reply template.
<code>user-mh-directory/replcomps</code>	The user's default reply template. (If it exists, it overrides the MH default reply template.)
<code>\$HOME/.mh-profile</code>	The MH user profile.
<code>user-mh-directory/draft</code>	The draft file.

## Related Information

Other MH commands: “**ali**” on page 48, “**anno**” on page 50, “**comp**” on page 185, “**dist**” on page 336, “**forw**” on page 438, “**mhl**” on page 643, “**prompter**” on page 778, “**send**” on page 893, “**whatnow**” on page 1215, “**whom**” on page 1222.

The **mh-alias**, **mh-format**, **mh-mail**, and **mh-profile** files in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

# restore

---

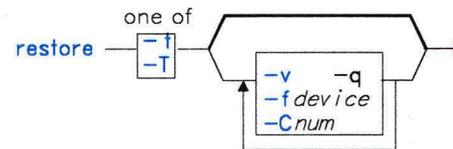
## restore

---

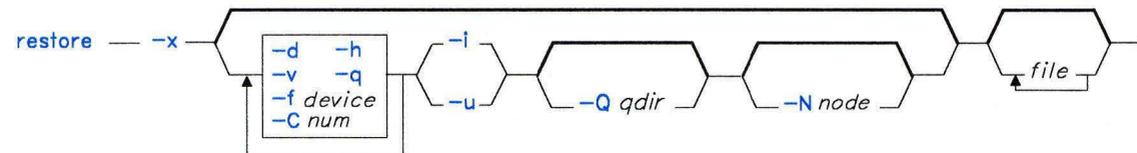
### Purpose

Copies back files created by the **backup** command.

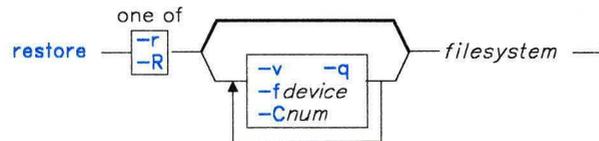
### Syntax



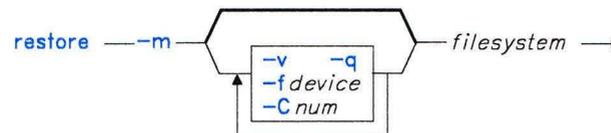
OL805251



OL805352



OL805353



OL805441

### Description

The **restore** command reads files written by the **backup** command to a backup medium and restores them to a file system. You can restore files that are backed up on a local system or on a remote system.

There are four ways to use the **restore** command:

- To display a table of contents for the backup (**-T**) or to display label information (**-t**)
- To restore specified files (**-x**)
- To restore an entire file system (**-r**) or begin at an arbitrary volume number (**-R**).
- To restore an entire minidisk (**-m**).

When you do not specify a restore device, the **restore** command reads files from a default backup device. For restore by name, **restore -x**, the system reads from **/dev/rfd0** unless you specify a device with the **-f** flag. For restore by file system, **restore -i**, or restore by minidisk, **restore -m**, if **/etc/filesystems** contains a stanza that matches the name you specified and a stanza with a **backupdev** entry, then the system reads from the device specified by **backupdev**. Otherwise, the system reads from **/dev/rmt0** or the *device* specified with the **-f** flag.

If neither **-i** nor **-u** is specified, files are restored to the local node in the current directory. If either **-i** or **-u** is specified, the system needs to know where to restore the files. Either a target node or a qualifying directory or both can be used to tell the system where to restore the files. If they are not specified, **restore** looks for the information in the header file of the backup. If you are restoring files backed up with an old version of the **backup** that does not contain a header and you do not specify a target node and a qualifying directory on the command line, the **restore** command ends in an error.

#### Notes:

1. If you restore by file system or by minidisk, the source and target must be on the local system. To restore from a remote system, restore by name with the **-i** or **-u** flags. These flags allow users in a distributed services environment to restore files backed up on a remote tape drive.
2. If the file system you are restoring is mounted and is not the root file system, **restore** unmounts the file system before it performs an i-node or a minidisk restore and then remounts the file system before quitting. If the file systems you are restoring include the root file system, **restore** ensures that the other file systems are not in use. If one is, it warns you of this and quits.
3. Files must be restored using the same method by which they were backed up. For example, if a file system was backed up by minidisk, it must be restored by minidisk.
4. When more than one diskette is required, **restore** reads the one mounted, prompts the user for a new one, and waits for the user's response (unless you are in unattended mode). After inserting the new diskette, press **Enter** to continue restoring files.

## Flags

- Cnum** Specifies the number of blocks to read in a single input operation. If you do not specify this flag, **restore** selects a default value appropriate for the physical device you have selected. Larger values of *num* result in longer physical transfers from tape devices. **restore** always ignores the value of the **-C** flag

when it reads a diskette; the input is always read in clusters that occupy a complete track.

- d** Indicates that if *file* is a directory, all files in that directory should be restored. In this case, the name of each restored file is always its name as shown by **restore -T**, whether the backup was by name or by i-node. The *file* names supplied need not be directories. Thus, for i-node backups:

```
restore -x a/b/file.c
```

creates a file whose name is its i-node number, while:

```
restore -xd a/b/file.c
```

creates a file named `a/b/file.c`. With this flag, *file* names can include pattern-matching characters, although you must quote these characters to prevent their expansion by the shell.

Use this flag only when you are restoring by individual file name (**-x**).

- fdevice** Specifies the input device. Specify *device* as a file name (such as `/dev/rmt0`) to get input from the named device or specify **-** (minus) to get input from the standard output device. The **-** feature enables you to improve performance when restoring from streaming tape by piping the output of a **dd** command to the **restore** command (see example). The **restore** command recognizes a special syntax for the names of input files. If the device parameter is a range of names, for example `/dev/rfd0-3`, **restore** automatically goes from one drive in the range to the next. After using all of the specified drives, it stops and requests that another diskette be inserted.
- h** Specifies that the access and modification times of restored files are to be set to the time of restoration. (The default action is to set the access and modification times to the file times on the backup medium.) If a restored file is an archive, the modification times in all the member headers are also set to the time of restoration. You can specify this flag only when you are restoring individually named files.
- i** Enables users in a distributed services environment to restore from a backup medium on a remote system in interactive mode (user input is permitted).
- m** Restores an entire minidisk as an exact image.  
**Note:** You can use this flag only with minidisks that are at least as large as the original minidisk that was backed up. If the minidisk is larger than the original, the leftover space becomes unusable after restoring the minidisk. You can use **restore -t** to see how large a minidisk you need.
- N node** Specifies the node on which to restore files. The *node* can be a node nickname or a node id. The **restore** command uses this node instead of the node in the backup header.

- q Specifies that the removable medium is ready to use. In this case, **restore** proceeds without prompting you to prepare the removable medium.
- Q *qdir* Specifies the qualifying directory in which to restore files. The *qdir* can be a relative or absolute directory. The **restore** command uses this qualifying directory instead of the directory in the backup header. Current directory relative names extracted from the backup medium are placed in this directory.
- r Restores an entire file system. Use this flag with i-node backups only (see “**backup**” on page 88). *filesystem* can be a device name (block or character special file) or a directory name that **restore** looks up in */etc/filesystems*.

If you are restoring a *full* (level 0) *backup*, run the **mkfs** command to create an empty file system before doing the restore. If you are restoring an *incremental backup* at, for example, level 2, run **mkfs**, then restore the appropriate level 0 backup, then the level 1 backup, and finally the level 2 backup.

**Warning:** If you do not follow this procedure carefully, you can ruin an entire file system. As an added safety precaution, run **fsck** after you restore each backup level.

- R Restarts an aborted **restore** at a specified point. **restore** prompts you for the starting volume number. This flag is invalid in combination with the **-m** flag.
- T Displays the backup file header and the names of the backed up files. If the backup was made by name (**backup -i**), the names displayed are the ones you provided to **backup**. If the backup was made by i-node, **restore** displays the i-number of each file along with the file name. The names are relative to the root directory of the file system backed up. The only exception is the root directory itself, whose name is given as a slash (/).
- t Displays only the backup file header.
- u Enables users to restore files in unattended mode (user input is not permitted) from a backup medium on a remote system. If any user input (such as `Please mount volume 1 on /dev/rfd0`) is required, the command ends in an error. This enables users to set up a shell file that restores files at night or at other times when a user is unavailable.
- v Reports the progress of the restoration as it proceeds.
- x Restores individually named files. The names must be in the same form as the names shown by **restore -T**. With a name backup, **restore** gives the restored file whatever name was supplied when the file was backed up. If the original name was specified relative to the current directory, **restore** creates a file relative to the current directory. **restore** automatically creates any needed directories. With an i-node backup, the name of the restored file is the same as its i-number. This flag is invalid with the **-m** flag and the **-r** flag.

### Examples

1. To list the names of files previously backed up:

```
restore -T
```

Information is read from the default backup device `/dev/rfd0`. If individual files were backed up, then only the file names are displayed. If an entire file system was backed up, the i-number is also shown.

2. To display technical information about a backup:

```
restore -t
```

This command displays information including when the backup was made, which file system was saved, and whether it is a backup by name, a backup by minidisk, or a backup by file system or i-node.

3. To restore files to the main file system:

```
restore -x -v
```

The `-x` extracts all the files from the backup medium and restores them to their proper places in the file system. The `-v` displays a progress report as each file is restored. If a file system backup is being restored, then the files are named with their i-numbers. Otherwise, just the names are displayed.

4. To copy selected files:

```
restore -xv /u/jim/manual/chap1
```

This command extracts the file `/u/jim/manual/chap1` from the backup medium and restores it. To work properly, `/u/jim/manual/chap1` must be a name that can be displayed by `restore -T`.

5. To copy all the files in a directory:

```
restore -xdv manual
```

This command restores the directory `manual` and the files in it. If it does not exist, a directory named `manual` is created in the current directory to hold the files being restored.

6. To restore an entire file system backup:

```
mkfs /dev/hd1  
restore -rv /dev/hd1
```

This command restores an entire file system backup onto `/dev/hd1`. It destroys and replaces any file system that was previously stored on `/dev/hd1`. If the backup was made using incremental file system backups, restore the backups in increasing backup-level order (0, 1, 2 . . . ).

7. To restore a minidisk:

```
restore -m /dev/hd1
```

This restores the exact image of minidisk /dev/hd1. You can also identify the minidisk by its stanza name in the `/etc/filesystems` file.

8. To restore files in interactive mode from the remote default device specifying a new target node and qualifying directory:

```
restore -xi -N nick -Q /u/nick
```

This command extracts the files from the default remote backup device and restores them to the node `nick`. Any unqualified names from the media are extracted relative to the directory `/u/nick`. The contents of the backup header (if a header exists) is ignored.

9. To restore files in unattended mode specifying a target node id:

```
restore -xuN 10813661
```

Files from the default backup device are restored at the node whose node ID is 10813661. Since a qualifying directory is not specified and the backup contains a backup header file, the **restore** command extracts the files to the qualifying directory specified in the header.

10. To improve performance on streaming tape, pipe the **dd** command to the **restore** command:

```
dd of=/dev/rmt0 bs=30b | restore -xf-
```

The **dd** command copies the files from an output file which is a streaming tape device (`of=/dev/rmt0`) and specifies a file size of 30 blocks (`bs=30b`). The output is piped to **restore**. The **restore** command gets the input from the standard input device (`f-`) and restores up by name (`x`).

## Files

<code>/etc/filesystems</code>	Descriptions of mountable file systems; consulted for default parameters.
<code>/dev/rfd0</code>	Default restore device.

## Related Information

The following command: “**backup**” on page 88.

The **file systems** and **backup files** in *AIX Operating System Technical Reference*.

“Backing up and Restoring Files” in *Using the AIX Operating System*.

# rex

---

## rex

---

### Purpose

Handles remote execution requests.

### Syntax

`/usr/etc/rpc.rex`

OL805507

### Description

When a remote execution request is made, the **inetd** daemon starts **rex** if the appropriate entry is in the **/etc/inetd.conf** file.

Noninteractive programs use standard file descriptors connected directly to Simulates phototypesetter output for a Tektronix 4014 work station connections. Interactive programs use pseudo-terminals, similar to the login sessions provided by **rlogin**. Diagnostic messages are normally displayed on the console and returned to the requestor.

The **rex** daemon can use NFS to mount file systems specified in the remote execution request.

---

#### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

### Files

<code>/dev/ptsn</code>	Pseudo-terminals used for interactive mode.
<code>/etc/passwd</code>	List of authorized users.
<code>/etc/inetd.conf</code>	TCP/IP configuration file.

### Related Information

The following command: “**on**” on page 726.

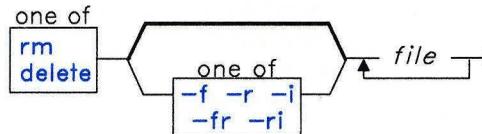
---

**rm**

---

**Purpose**

Removes files or directories.

**Syntax**

OL805212

**Description**

The **rm** (**delete**) command removes the entries for *files* from a directory. If an entry is the last link to a file, it is destroyed. To remove a file, you must have write permission in its directory, but neither read nor write permission for the file itself if you own it or are acting with superuser authority.

If a file has no write permission and standard input is a work station, **rm** displays the file permission code and reads a line from standard input. If that line begins with *y*, **rm** deletes the file. If the response is anything other than *y*, **rm** does nothing.

---

**Japanese Language Support Information**

An affirmative response in Japanese Language Support matches one of the elements in the environment variable **YESSTR**.

---

**Flags**

- f** Does not prompt before removing a write-protected file.
- i** Prompts you before deleting each file. When you use both **-i** and **-r** together, **rm** also asks if you want to examine directories.
- r** Permits recursive removal of directories and their contents (for cases where *file* is a directory).

## Examples

1. To delete a file:

```
rm myfile
```

If there is another link to this file, then the file remains under that name, but the name `myfile` is removed. If `myfile` is the only link, the file itself is deleted.

2. To delete a file silently:

```
rm -f core
```

This removes `core` without asking any questions or displaying any error messages. This is normally used in shell procedures. It prevents confusing messages from being displayed when deleting files that may or may not exist.

3. To delete files one by one:

```
rm -i mydir/*
```

This interactively asks you if you want to remove each file. After each file name is displayed, enter `y` to delete the file, or press **Enter** to keep it.

---

### Japanese Language Support Information

Enter one of the allowed affirmative responses at the prompts. The allowed affirmative responses are defined in the environment variable **YESSTR**.

---

4. To delete a directory tree:

```
rm -ir manual
```

This recursively removes the contents of all subdirectories of `manual`, then removes `manual` itself, asking if you want to remove each file. For example:

```
You: rm -ir manual
System: directory manual:
You: y
System: directory manual/draft1:
You: y
System: manual/draft1/chapter1:
You: y
System: manual/draft1/chapter2:
You: y
System: manual/draft1:
You: y
System: directory manual/draft2
```

```
You: y
System: manual/draft2:
You: n
System: manual:
You: y
```

Here, **rm** first asks if you want it to search the directory `manual`. Because `manual` contains directories, **rm** next asks for permission to search `manual/draft1` for files to delete, and then asks if you want it to delete the files `manual/draft1/chapter1` and `manual/draft1/chapter2`. **rm** next asks for permission to search the directory `manual/draft2`, and then asks for permission to delete the directories `manual/draft1`, `manual/draft2`, and `manual`. Because you denied permission to remove `manual/draft2`, **rm** will not remove `manual`. Instead, you will see the message `rmdir: manual not empty`.

## Related Information

The following commands: “**del**” on page 308 and “**ln**” on page 581.

The **unlink** system call in *AIX Operating System Technical Reference*.

The discussion of Japanese Language Support in *Japanese Language Support User's Guide*.

# rmail

---

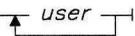
## rmail

---

### Purpose

Handles remote mail received via uucp.

### Syntax

rmail 

AJ2FL255

### Description

The **rmail** command interprets incoming mail received via **uucp**. It collapses FROM lines in the form generated by **bellmail** into a single line of the form:

*return-path!sender*

It passes the processed mail on to **sendmail**.

The **rmail** command works with **uucp** and **sendmail**. This is a new command with Version 2.2 of AIX Operating System. It is not the same as the **rmail** command found in earlier versions of AIX Operating System.

### Related Information

The following commands: “**uucp**” on page 1144 and “**sendmail**” on page 897.

---

# rmdel

---

## Purpose

Removes a delta from a Source Code Control System (SCCS) file.

## Syntax

```
rmdel — -rSID file
```

OL805213

## Description

The **rmdel** command removes the delta specified by *SID* from each named **Source Code Control System** (SCCS) *file*. You can remove only the most recently created delta in a branch, or the latest trunk delta if it has no branches. In addition, the *SID* you specify must not be a version currently being edited for the purpose of making a delta. To remove a delta, you must either own the SCCS file and the directory, or you must be the user who created the delta you want to remove.

If you specify a directory in place of *file*, **rmdel** performs the requested actions on all SCCS files (those with file names that have the *s.prefix*). If you specify a - (minus) in place of *file*, **rmdel** reads standard input, and interprets each line as the name of an SCCS file. **rmdel** continues to take input until it reads an end-of-file character.

## Flag

**-rSID** Removes the delta *SID* from the SCCS file. This flag is required.

## Related Information

The following commands: “**delta**” on page 310, “**get**” on page 477, “**help**” on page 513, and “**prs**” on page 781.

The **sccsfile** file in *AIX Operating System Technical Reference*.

The discussion of SCCS in *AIX Operating System Programming Tools and Interfaces*.

# rmdir

---

## rmdir

---

### Purpose

Removes a directory.

### Syntax

```
rmdir directory
```

OL805252

### Description

The **rmdir** command removes a *directory* from the system. The *directory* must be empty before you can remove it, and you must have write permission in its parent directory. Use the **ls -l** command to see if the *directory* is empty.

### Example

To empty and remove a directory:

```
rm mydir/* mydir/.  
rmdir mydir
```

This removes the contents of `mydir`, then removes the empty directory. The **rm** command displays an error message about trying to remove the directories `.` (dot) and `..` (dot dot), and then **rmdir** removes them.

Note that `rm mydir/* mydir/.*` first removes files with names that do not begin with a dot, then those with names that do begin with a dot. You may not realize that the directory contains file names that begin with a dot because the **ls** command does not normally list them.

### Related Information

The following command: “**rm**” on page 833.

The **unlink** and **rmdir** system calls in *AIX Operating System Technical Reference*.

---

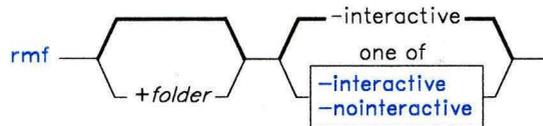
# rmf

---

## Purpose

Removes a folder.

## Syntax



rmf — -help —

AJ2FL165

## Description

The **rmf** command is used to remove folders and the messages that they contain. **rmf** is part of the MH (Message Handling) package and can be used with other MH and AIX commands.

**Warning:** The **rmf** command irreversibly deletes messages that do not have other links.

The **rmf** command deletes all of the messages within the specified folder and then deletes the folder. If the folder contains files that are not messages, **rmf** does not delete those files and reports an error.

If you have read-only access to the specified folder, **rmf** does not delete the folder or any of its messages. **rmf** deletes only your private sequences and your current message information from the profile.

The **rmf** command does not delete folders recursively. Thus, you cannot remove subfolders by requesting the removal a parent folder.

## rmf

---

### Flags

- +folder** Specifies the folder to be removed. If you remove a subfolder, the parent of that folder becomes the current folder. If you remove the current folder, **+inbox** becomes current. The default folder is the current folder. If **+folder** is not specified and **rmf** cannot find the current folder, **rmf** requests confirmation for removing **+inbox**.
- help** Displays help information for the command.
- interactive** Requests confirmation before removing the folder. If **+folder** is not specified, this is the default.
- nointeractive** Removes the folder and its messages without requesting confirmation. If **+folder** is specified, this is the default.

### Profile Entries

- Current-Folder:** Sets your default current folder.
- Path:** Specifies your *user-mh-directory*.

### Files

- `$HOME/.mh-profile` The MH user profile.

### Related Information

- The MH command “**rmm**” on page 841.
- The **mh-profile** file in *AIX Operating System Technical Reference*.
- “Overview of the Message Handling Package” in *Managing the AIX Operating System*.

---

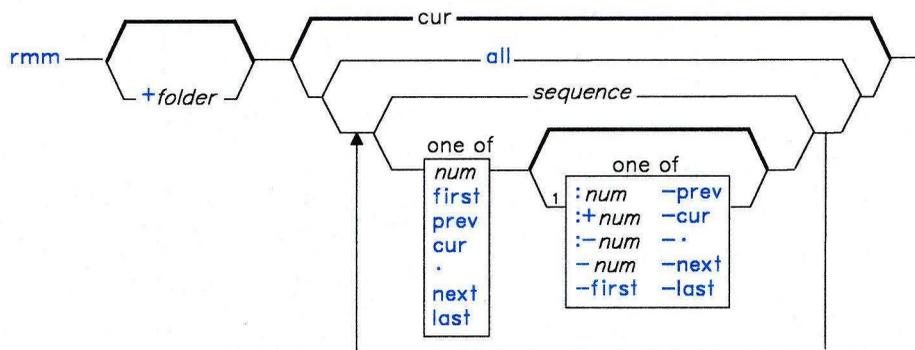
## rmm

---

### Purpose

Removes messages.

### Syntax



rmm — -help —

AJ2FL202

AJ2FL203

<sup>1</sup> Do not put a blank between these items.

OL805308

### Description

The **rmm** command is used to remove messages from active status. **rmm** is part of the MH (Message Handling) package and can be used with other MH and AIX commands.

The **rmm** command renames the specified message files so that their file names have preceding commas. You can use these files as temporary backups and arrange for the **cron** command to delete your backups periodically.

## rmm

---

### Flags

**+folder msgs** Specifies the messages that you want to remove. *msgs* can be several messages, a range of messages, or a single message. You can use the following message references when specifying *msgs*:

<i>num</i>	<b>first</b>	<b>prev</b>
<b>cur</b>	.	<b>next</b>
<b>last</b>	<b>all</b>	<i>sequence</i>

The default message is the current message in the current folder. **rmm** does not change the current message.

**-help** Displays help information for the command.

### Profile Entries

<b>Current-Folder:</b>	Sets your default current folder.
<b>Path:</b>	Specifies your <i>user_mh_directory</i> .
<b>rmmproc:</b>	Specifies the program used to remove messages from a folder.

### Files

`$HOME/.mh_profile` The MH user profile.

### Related Information

The MH command “**rmf**” on page 839.

The **mh-profile** file in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

---

## rpcgen

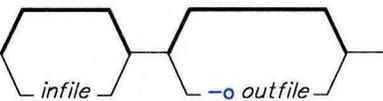
---

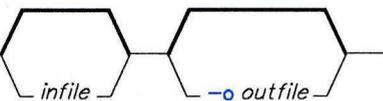
### Purpose

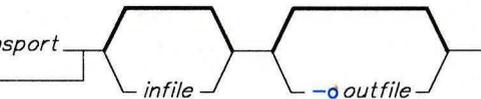
Compiles a Remote Procedure Call program.

### Syntax

`rpcgen infile`

`rpcgen -c` 

`rpcgen -h` 

`rpcgen -s transport` 

OL805497

### Description

The **rpcgen** command generates C Language code for implementing an RPC protocol. Input to **rpcgen** is in Remote Procedure Call Language (RPCL). RPCL is similar to the C Programming Language.

The **rpcgen** command operates in the following modes:

- Converts RPCL definitions to C Language definitions and puts them in a header file.
- Compiles the XDR routines that serialize or convert the data between the machine issuing the Remote Procedure Call and the machine carrying it out.
- Compiles converted RPCL definitions and puts them in a header file named *infile.h*. Compiles the XDR routines and puts them in *infile.c*.

## rpcgen

---

- Compiles an RPC server skeleton. Using the skeleton, you can write local procedures that implement RPC servers without invoking RPC protocols.

In each mode, the input can contain comments (with the same format as C Language comments) and preprocessor directives. The comments are ignored and the directives are copied into the output header file. You can customize XDR routines by leaving some data types undefined. For every undefined data type, **rpcgen** will assume that a routine with an **xdr-** prefix exists.

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

## Flags

- |                            |   |
|----------------------------|---|
| <b>-c</b>                  | Compiles XDR routines.  |
| <b>-h</b>                  | Compiles C data definitions in a header file.   |
| <b>-o <i>outfile</i></b>   | Specifies the name of the output file. If you do not specify an output file, <b>rpcgen</b> uses the standard output as the default.                       |
| <b>-s <i>transport</i></b> | Compiles a server using a specified data transport. This flag can be invoked more than once in order to compile a server that serves multiple transports. |

---

# rpcinfo

---

## Purpose

Reports Remote Procedure Call status information

## Syntax

```
/etc/rpcinfo -p host
```

```
/etc/rpcinfo -u host program version
```

```
/etc/rpcinfo -t host program version
```

OL805495

## Description

The **rpcinfo** command reports the status of Remote Procedure Call services on a specified host. The *host* parameter specifies the Remote Procedure Call server. The *program* parameter specifies the program used by the remote procedure. It can be a name or a number. The version number of the program used by the remote procedure is specified by *version*. If you do not specify a version number, **rpcinfo** searches for all registered version numbers and calls each one.

**Note:** The **rpcinfo** command uses *version 0* to search for all registered versions, since 0 is not usually assigned as a program's version number. If *version 0* is assigned to a program's version number, **rpcinfo** uses a high level number in its place.

## rpcinfo

---

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

### Flags

- p** Queries the *host* portmap service and displays a list of registered RPC programs. If *host* is not specified, the value returned by **hostname** is the default value.
- t** Uses the TCP/IP data transport to make a Remote Procedure Call to **procedure** 0 of the *program* on the specified *host* and report if a response was received.
- u** Use the UDP/IP data transport to make a Remote Procedure Call to **procedure** 0 of the *program* on the specified *host* and report if a response was received.

### Files

`/etc/rpc` RPC program names.  
`/etc/inetd.conf` TCP/IP configuration file.

## **rstatd**

---

### **Purpose**

Returns NFS performance statistics from the kernel

### **Syntax**

`/usr/etc/rpc.rstatd` —|

OL805494

### **Description**

The **rstatd** daemon returns performance statistics from the kernel. The **inetd** daemon invokes **rstatd**.

---

#### **Japanese Language Support Information**

If Japanese Language Support is installed on your system, this command is not available.

---

### **File**

`/etc/inetd.config` TCP/IP configuration file.

### **Related Information**

The following command: “**rup**” on page 854.

# runacct

---

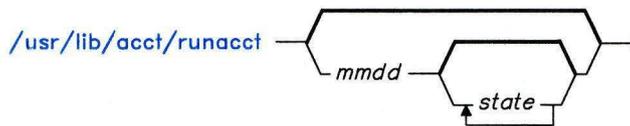
## runacct

---

### Purpose

Runs daily accounting.

### Syntax



OL805253

### Description

The **runacct** command is the main daily accounting shell procedure. Normally initiated by **cron**, **runacct** processes connect, fee, disk, queueing system and process accounting data files. It also prepares summary files for the **prdaily** procedure or for billing purposes.

The **runacct** command protects active accounting files and summary files in the event of run-time errors. It records its progress by writing descriptive messages into the file **/usr/adm/acct/nite/active**. When **runacct** encounters an error, it writes a diagnostic message to **/dev/console**, sends **mail** to users **root** and **adm**, and exits.

The **runacct** procedure also creates two temporary files, **lock** and **lock1** in the directory **/usr/adm/acct/nite**, which it uses to prevent two simultaneous calls to **runacct**. It uses the file **lastdate** (in the same directory), to prevent more than one invocation per day.

The **runacct** command breaks its processing into separate, restartable **states**. As it completes each state, it writes the name of the next state in **/usr/adm/acct/nite/statefile**. **runacct** processes the various states in the following order:

State	Actions
SETUP	Moves the active accounting files to working files and restarts the active files.
WTMPFIX	Verifies the integrity of the <b>wtmp</b> file, correcting date changes if necessary.
CONNECT1	Calls <b>acctcon1</b> to produce connect session records.

---

CONNECT2	Converts connect session records into total accounting records ( <b>tacct.h</b> format).
PROCESS	Converts process accounting records into total accounting records ( <b>tacct.h</b> format).
MERGE	Merges the connect and process total accounting records.
FEES	Converts the output of <b>chargefee</b> into total accounting records ( <b>tacct.h</b> format) and merges them with the connect and process total accounting records.
DISK	Merges disk accounting records with connect, process, and fee total accounting records.
QUEUEACCT	Sorts the queue (printer) accounting records, converts them into total accounting records ( <b>tacct.h</b> format), and merges them with other total accounting records.
MERGETACCT	Merges the daily total accounting records in <b>daytacct</b> with the summary total accounting records in <b>/usr/adm/acct/sum/tacct</b> .
CMS	Produces command summaries in the file <b>/usr/adm/acct/sum/cms</b> .
USEREXIT	If the shell file <b>/usr/adm/siteacct</b> exists, calls it at this point to perform site-dependant processing.
CLEANUP	Deletes temporary files and exit.

To restart **runacct** after a failure, first check the **/usr/adm/acct/nite/active** file for diagnostic messages, then fix any damaged data files such as **pacct** or **wtmp**. Remove the **lock** files and **lastdate** file (all in the **/usr/adm/acct/nite** directory), before restarting **runacct**. You must specify the **mmdd** parameter if you are restarting **runacct**. It specifies the month and day for which **runacct** is to rerun the accounting. **runacct** determines the entry point for processing by reading **statefile**. To override this default action, specify the desired **state** on the **runacct** command line. For a more detailed discussion of restarting **runacct**, see *Managing the AIX Operating System*.

It is not usually a good idea to restart **runacct** in the **SETUP state**. Instead, perform the setup actions manually and restart accounting with the **WTMPFIX** state, as follows:

```
runacct mmdd WTMPFIX
```

If **runacct** fails in the **PROCESS** state, remove the last **ptacct** file, because it will be incomplete.

# runacct

---

## Japanese Language Support Information

This command has not been modified to support Japanese characters.

---

## Examples

1. To start **runacct**:

```
nohup /usr/lib/acct/runacct 2> /usr/adm/acct/nite/accterr &
```

This starts **runacct** in the background (&), ignoring all INTERRUPT and QUIT signals (**nohup**). All standard error output is written to the file `/usr/adm/acct/nite/accterr`.

2. To restart **runacct**:

```
nohup /usr/lib/acct/runacct 0601 2>> /usr/adm/acct/nite/accterr &
```

This restarts **runacct** for the day of June 1 (0601). **runacct** reads the file `/usr/adm/acct/nite/statefile` to find out the state to begin with. Standard error output is added to the end of the file `/usr/adm/acct/nite/accterr`.

3. To restart **runacct** in a specific state, in this case the **MERGE** state:

```
nohup /usr/lib/acct/runacct 0601 MERGE 2>> /usr/adm/acct/nite/accterr &
```

## Files

<code>/usr/adm/wtmp</code>	Login/logoff history file.
<code>/usr/adm/pacct*</code>	Process accounting file.
<code>/usr/adm/acct/nite/daytacct</code>	Disk usage accounting file.
<code>/usr/adm/qacct</code>	Active queue accounting file.
<code>/usr/adm/fee</code>	Record of fees charge to users.
<code>/usr/adm/acct/sum/*</code>	Command and total accounting summary files.
<code>/usr/adm/acct/nite/ptacct*.mmd</code>	Concatenated version of <b>pacct</b> files.
<code>/usr/adm/acct/nite/active</code>	<b>runacct</b> message file.
<code>/usr/adm/acct/nite/lock*</code>	Prevent simultaneous invocation of <b>runacct</b> .
<code>/usr/adm/acct/nite/lastdate</code>	Contains last date <b>runacct</b> was run.
<code>/usr/adm/acct/nite/statefile</code>	Contains current state to process.

## Related Information

The following commands: “**acct/\***” on page 13, “**acctcms**” on page 18, “**acctcom**” on page 20, “**acctcon**” on page 24, “**acctmerg**” on page 28, “**acctprc**” on page 30, “**cron**” on page 220, and “**fwtmp**” on page 457.

The **acct** system call and the **acct** and **utmp** files in *AIX Operating System Technical Reference*.

“Running System Accounting” in *IBM RT Managing the AIX Operating System*.

## runcat

---

## runcat

---

### Purpose

Pipes data from **mkcatdefs** to **runcat**

### Syntax



OL805491

### Description

The **runcat** command invokes the **mkcatdefs** command and pipes the message source data (the output from **mkcatdefs**) to the **gencat** program.<sup>8</sup> This method is simpler than using the redirection operator **>** to capture the output from **mkcatdefs** and then running **gencat**. The format for **runcat** is:

```
$ runcat catname sourcefile [catfile]
```

The file *sourcefile* contains the message text with your symbolic identifiers. **mkcatdefs** uses *catname* to generate the name of the symbolic definition file by adding **-msg.h** to the end of *catname*, and to generate the symbolic name for the catalog file by adding **MF-** to the beginning of *catname*. The definition file must be included in your application program. The symbolic name for the catalog file can be used in **catopen** or **NLcatopen** instead of the actual file name.

*catfile* is the name of the catalog file created by **gencat**. If you do not specify this parameter, **gencat** names the catalog file by adding **.cat** to the end of *catname*. This file name can also be used in **catopen** or **NLcatopen**.

---

#### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

---

<sup>8</sup> **runcat** is an AIX extension to the X/Open standard.

## Related Information

The following commands: “**dspcat**” on page 357, “**dspmsg**” on page 359, “**gencat**” on page 470, and “**mkcatdefs**” on page 651.

The **catopen**, **catgets**, **catgetmsg**, **catclose**, **NLcatopen**, **NLcatgets**, and **NLgetmsg** files in *AIX Operating System Technical Reference*.

The discussion of **runcat** in *AIX Operating System Programming Tools and Interfaces*.

## rup

---

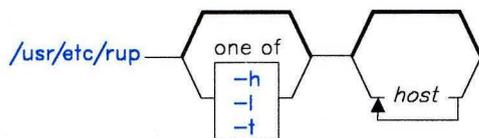
## rup

---

### Purpose

Displays the status of hosts on a network.

### Syntax



OL805492

### Description

The **rup** command broadcasts a query on the local network and displays the responses it receives. It gives a status report on system usage times (uptimes) and load averages.

To query specific hosts, list their names as arguments following **rup**.

**Note:** Remote hosts respond only if running the **rstatd** daemon. The **rstatd** daemon is started from **inetd**.

---

#### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

### Flags

When used without flags, **rup** displays the responses in the order it receives them. Use the following flags to change the display order.

- h** Displays responses alphabetically by host name.
- l** Displays responses by load average.
- t** Displays responses by up time.

## Related Information

The following command: “**rstatd**” on page 847.

## rusers

---

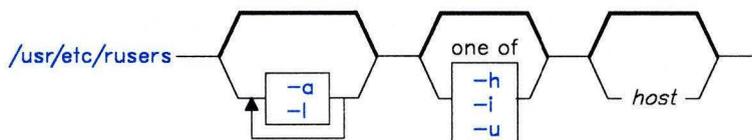
## rusers

---

### Purpose

Identifies users logged in on network hosts

### Syntax



OL805496

### Description

The **rusers** command broadcasts a query on the local network and displays the responses it receives.

To query specific hosts, list their names following the **rusers** command.

**Note:** Remote hosts will only respond if they are running the **rusersd** daemon (started from `/etc/inetd.conf`).

---

#### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

### Flags

The **rusers** command displays the responses to the query in the order it receives them. Each machine is listed on a separate line. Use the following flags to change the print order:

- a Responds for all machines even if no users are logged in.
- h Sorts responses alphabetically by host name.
- i Sorts responses by idle time. Idle time is reported if a user has not typed into the system for more than a minute.
- l Responds with a longer listing in the style of the **who** command.

**-u**       Sorts responses by numbers of users.

## **File**

`/etc/inetd.conf`   TCP/IP configuration file.

## **Related Information**

The following commands: “**who**” on page 1219 and “**rusersd**” on page 858.

## **rusersd**

---

## **rusersd**

---

### **Purpose**

Responds to queries from **rusers** command.

### **Syntax**

`/usr/etc/rpc.rusersd`

OL805505

### **Description**

The **rusersd** daemon allows remote hosts to respond to queries from the **rusers** command. The **inetd** daemon invokes **rusersd**.

---

#### **Japanese Language Support Information**

If Japanese Language Support is installed on your system, this command is not available.

---

### **File**

`/etc/inetd.conf` TCP/IP configuration file.

### **Related Information**

The following command: “**rusers**” on page 856.

---

# rwall

---

## Purpose

Writes to all users over a network.

## Syntax

```
/usr/etc/rwall — -h host — -n netgroup —|
```

```
/usr/etc/rwall — -n netgroup —|
```

```
/usr/etc/rwall — host —|
```

OL805547

## Description

The **rwall** command reads a message from standard input and broadcasts it to all users logged in to the specified host machines. It reads the message from standard input until it reaches an **end-of-file** character.

The **rwall** sends the message with the following introduction line:

Broadcast Message.....

**Note:** Users can only receive a message if they are running **rwalld**. This daemon is started by the **inetd** daemon.

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

## Flags

**-n** Sends the message to specific network groups only. Network groups are defined in the **netgroup** file.

# rwall

---

## File

*/etc/netgroup*

## Related Information

The following command: “**rwalld**” on page 861.

The **netgroup** file format in AIX Operating System Technical Reference.

---

# rwalld

---

## Purpose

Handles requests for the **rwall** and **shutdown** commands.

## Syntax

`/usr/etc/rwalld` —|

OL805545

## Description

The **rwalld** daemon handles requests from the **rwall** and **shutdown** commands. The **inetd** daemon invokes **rwalld**.

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

## File

`/etc/inetd.conf` TCP/IP configuration file.

## Related Information

The following commands: “**rwall**” on page 859 and “**shutdown**” on page 946.

**sact**

---

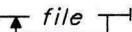
**sact**

---

## Purpose

Displays current Source Code Control System (SCCS) file editing status.

## Syntax

`sact` 

OL805063

## Description

The `sact.` command reads **Source Code Control System** (SCCS) files and writes to standard output the contents, if any, for the *p-file* associated with *file* (see “SCCS Files” on page 478 for information on the contents of the p-file). If - (minus) is specified for *file*, `sact.` reads standard input, and interprets each line as the name of an SCCS file. If *file* is a directory, `sact.` performs its actions on all SCCS files (that is, those files with the *s.* prefix).

---

### Japanese Language Support Information

This command has not been modified to support Japanese characters.

---

## Related Information

The following commands: “**delta**” on page 310, “**get**” on page 477, and “**unget**” on page 1116.

The `sccsfile` file in *AIX Operating System Technical Reference*.

The discussion of SCCS in *AIX Operating System Programming Tools and Interfaces*.

---

## sadc

---

### Purpose

Provides a system activity report package.

### Syntax

```

/usr/lib/sa/sadc { interval — num } { outfile }
/usr/lib/sa/sa1 { interval — num }
/usr/lib/sa/sa2 1

```

---

<sup>1</sup> See the **sar** command for the format and flag description. Note that you cannot use the **-o** and **-f** flags with **sa2**.

OL805254

### Description

The operating system contains a number of counters that are incremented as various system actions occur. They include the following:

- System unit utilization counters
- Buffer usage counters
- Disk and tape I/O activity counters
- **tty** device activity counters
- Switching and system-call counters
- File-access counters
- Queue activity counters
- Interprocess communications counters

The **sadc** command and the **sa1** and **sa2** shell procedures sample, save, and process this data.

**Note:** These commands only report on local activities.

### sadc

The **sadc** command, the data collector, samples system data *num* times every *interval* seconds. It writes in binary format to *outfile* or to the standard output. If you do not specify *interval* or *num*, a special record is written. This facility is used at system startup to mark the time when the counter restarts from zero.

### sa1

Use the shell procedure **sa1**, a variant of **sadc** to collect and store binary data in the file `/usr/adm/sa/sadd`, where *dd* is the day of the month. The *interval* and *num* parameters specify that the record should be written *num* times at *interval* seconds. If you do not specify these parameters, one record is written. You must have permission to write in the directory `/usr/adm/sa` to use this command.

The **sa1** command is designed to be started automatically by the **cron** command.

---

### Japanese Language Support Information

This command has not been modified to support Japanese characters.

---

### sa2

Use the shell procedure **sa2**, a variant of the **sar** command, to write a daily report in the file `/usr/adm/sa/saradd`. See “**sar**” on page 867 for a description of the flags.

The **sa2** command is designed to be started automatically by the **cron** command.

---

### Japanese Language Support Information

This command has not been modified to support Japanese characters.

---

## Files

<code>/usr/adm/sa/sadd</code>	Daily data file, <i>dd</i> represents the day of the month.
<code>/usr/adm/sa/saradd</code>	Daily report file, <i>dd</i> represents the day of the month.
<code>/tmp/sa.adrfl</code>	Address file.

## Related Information

The following commands: “**cron**” on page 220, “**sag**” on page 865, “**sar**” on page 867, and “**timex**” on page 1069.

---

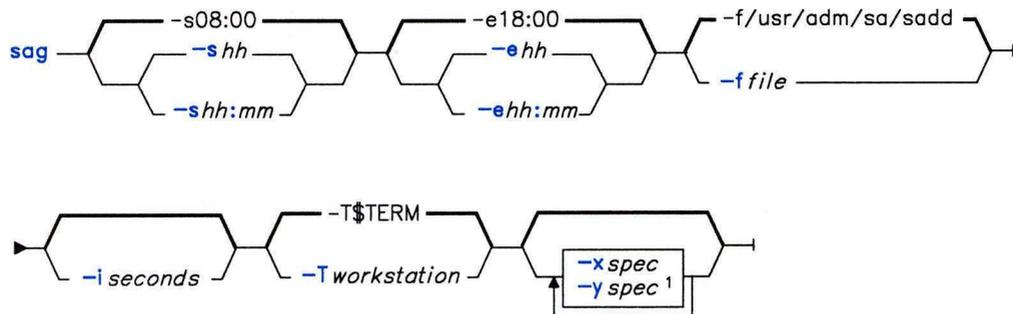
## sag

---

### Purpose

Displays a graph of system activity.

### Syntax



<sup>1</sup> The default for `-y` is `'%usr 0 100; %usr + %sys 0 100; %usr + %sys + %wio 0 100'`

OL805387

### Description

The **sag** command displays a graph of system activity. It gets information either from the daily activity file **usr/adm/sa/sadd** or from the binary data file selected by the **-f** flag. You must have already created this file by running the **sar** command with the **-o** flag. (See “**sar**” on page 867.)

The **sag** command calls the **sar** command, selecting the desired data by string-matching the data column header.

### Flags

The **sag** command passes the first four of the following flags to **sar** in order to collect the desired data for display. The last three flags specify plotting parameters.

**-e hh[:mm]** Selects data up to the time specified by **hh[:mm]**. The default time is 18:00.

- f** *file* Reads data from *file*. The default *file* is `/usr/adm/sa/sadd`, the current daily data file.
- i** *seconds* Selects data at intervals as close as possible to *seconds*.
- s** *hh[:mm]* Selects data later than the specified time. Default is 08:00.
- T** *workstation* Produces output suitable for *workstation*. (See “**tplot**” on page 1079 for known work stations.) If you do not specify a work station, **sag** uses the value found in the shell variable **\$TERM**.
- x** *spec* Specifies the x axis. *spec* has the following form:

*name* [*opname*] . . . [*lo hi*]

where *name* is a character string matching a column header in the **sar**-created data file (with an optional device name in brackets), or it is an integer value. *op* is +, -, \*, or / surrounded by blanks, with up to five *names* specified. Parentheses are not recognized and evaluation is left to right. Note that + and - have precedence over \* and / in evaluating expressions. *lo* and *hi* specify numeric scale limits. If these limits are unspecified, **sag** gets these limits from the data.

- y** *spec* Specifies the y axis. *spec* has the same form as **x** *spec*.

Specify only one *spec* for the x axis. If unspecified, the x axis assumes the time specified with the **-e** and **-s** flags (or their defaults if they are not used) as x axis limits. You can specify up to five *specs* separated by : (semicolons) for **-y**. If unspecified, the y axis has the value:

```
-y "%usr 0 100; %usr + %sys 0 100; %usr + %sys + %wio 0 100"
```

If you include blanks or an escaped carriage return (**\Enter**) within the **-x** and **-y** *specs*, enclose them in " " (double quotation marks).

## Files

`/usr/adm/sa/sadd` Daily data file for day *dd*.

## Related Information

The following commands: “**sar**” on page 867 and “**tplot**” on page 1079.

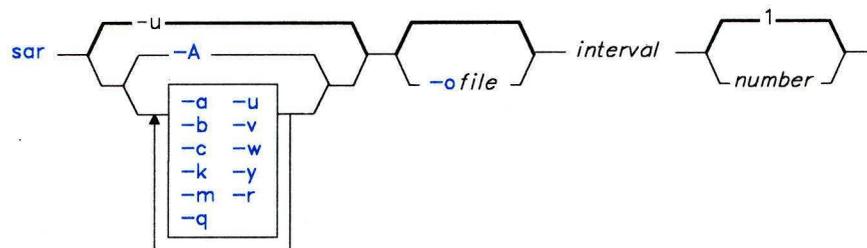
---

**sar**

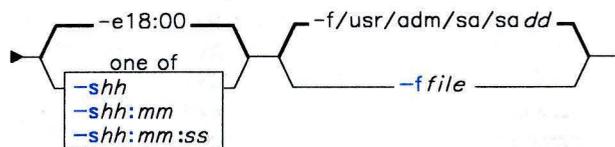
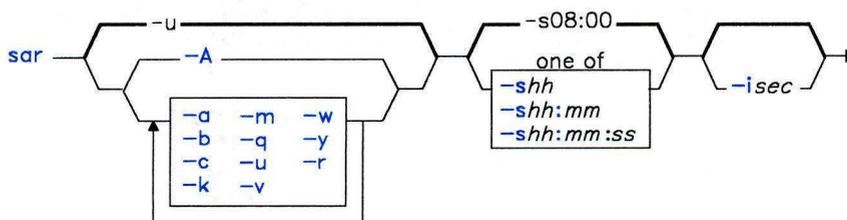

---

**Purpose**

Collects, reports, or saves system activity information.

**Syntax**


OL805390



OL805369

**Description**

The first format of the `sar` command writes to standard output the contents of selected cumulative activity counters in the operating system. It writes information a total of *number* times spaced *interval* seconds apart. The default value of *number* is 1. You can also save the collected data in the file specified by `-o file`.

In the second format (with no sampling interval specified), **sar** extracts and writes to standard output records previously saved in a file. This file can be either the one specified by the **-f** flag or, by default, the standard system activity daily data file, **/usr/adm/sa/sadd**, for the current day, *dd*.

You can select information about specific system activities with flags. Not specifying any flags selects only **cpu** activity. Specifying the **-A** flag selects all activities.

**Note:** This command only reports on local activities.

## Flags

- a** Reports use of file access system routines:
  - iget/s Calls per second to the i-node look-up routine.
  - namei/s Calls per second to the directory search routine.
  - dirblk/s Directory blocks read per second by namei().
- A** Report all data.
- b** Reports buffer activity for transfers, accesses, and cache hit ratios:
  - lread/s, lwrit/s Number of logical read/write requests per interval.
  - bread/s, bwrit/s Number of block read/write operations per interval.
  - %rcache, %wcache Cache hit ratios (for example, 1 - bread/lread).
  - pread/s, pwrit/s Read/writes per interval on seekable raw devices.
- c** Reports system calls:
  - scall/s Total number of system calls per second.
  - rchar/s, wchar/s Characters transferred per interval by read/write calls.
  - sread/s, swrit/s
  - fork/s, exec/s Specific system calls per second.
- e** *hh[:mm[:ss]]* Sets the ending time of the report. The default ending time is 18:00.
- f** *file* Extracts records from *file* (created by **-o file**). The default *file* is the current daily data file, **/usr/adm/sa/sadd**.
- i** *seconds* Selects data records at intervals as close as possible to the specified number of *seconds*. Otherwise, **sar** reports all intervals found in the data file.
- k** Reports kernel activity:
  - ksched/s Number of kernel processes assigned to tasks per second.
  - kproc-ov/s Number of overflows occurring between sampling points.
  - kexit/s Number of kernel processes terminating per second.
- m** Reports message and semaphore activities:
  - msg/s IPC message primitives per second.
  - sema/s IPC semaphore primitives per second.

- 
- o** *file* Saves the readings in *file* in binary form. Each reading is in a separate record and each record contains a tag identifying the time of the reading.
- q** Reports average queue length while occupied, and percentage of time occupied:  
runq-sz, %runocc Runs queue of processes in memory and runnable.
- r** Reports VRM paging statistics:  
slots The number of free pages on the paging minidisk.  
cycle/s The number of page replacement cycles per second.  
fault/s The number of page faults per second.  
odio/s The number of nonpaging disk I/Os per second.
- s** *hh[:mm[:ss]]* Sets the starting time of the data. That is, extract records time-tagged at or following the time specified. The default starting time is 08:00.
- u** Reports CPU activity (this flag is on by default):  
%usr Percentage of CPU time devoted to the user.  
%sys Percentage of CPU time devoted to the kernel.  
%wio Percentage of CPU time waiting for block I/O to complete.  
%idle Percentage of CPU time idle.
- v** Reports status of text, process, i-node, and file tables:  
text-sz, proc-sz, inod-sz, file-sz Entries in use at each sample point for each table.  
text-ov, proc-ov, inod-ov, file-ov Overflows occurring at each sample point for each table.
- w** Reports system switching activity:  
pswch/s Process switches per second.
- y** Reports TTY device activity:  
rawch/s TTY raw input queue characters per second.  
canch/s TTY canonical input queue characters per second.  
outch/s TTY output queue characters per second.  
revin/s TTY receive interrupts per second.  
xmtin/s TTY transmit interrupts per second.  
mdmin/s TTY modem interrupts per second.

## Files

- /usr/adm/sa/sadd* Daily data file, where *dd* are numbers representing the day of the month.

## **Related Information**

The following command: “**sag**” on page 865.

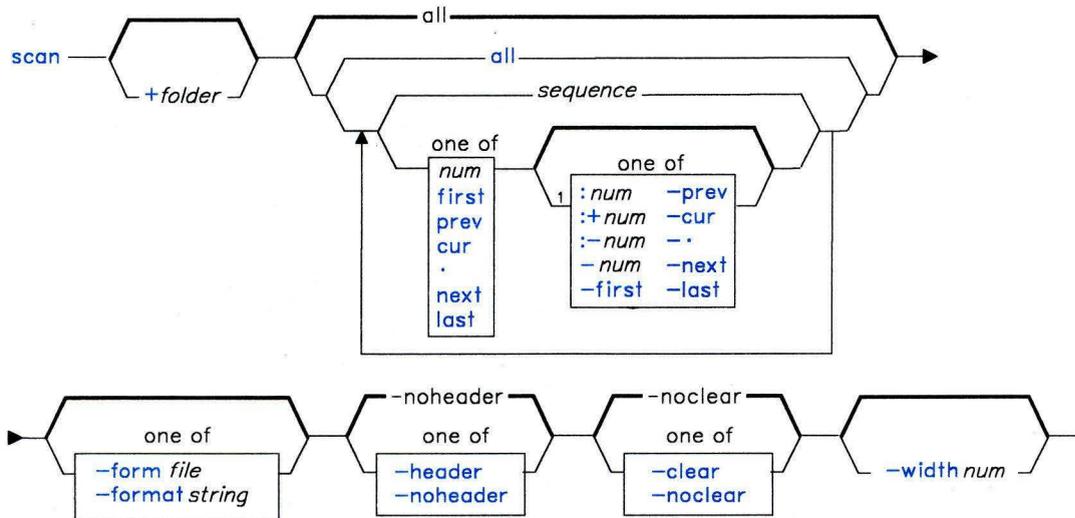
The discussion of monitoring system activity in *Managing the AIX Operating System*.

## scan

### Purpose

Produces a one line per message scan listing.

### Syntax



AJ2FL204

`scan` — `-help` —

AJ2FL206

<sup>1</sup> Do not put a blank between these items.

OL805308

### Description

The **scan** command is used to display information about the messages in a specified folder. **scan** is part of the MH (Message Handling) package and can be used with other MH and AIX commands.

## scan

---

The **scan** command displays a line of information about each specified message in the specified folder. Each line gives the message number, the date, the sender, the subject, and as much of the message body as possible. If a + symbol is displayed after the message number, the message is the current message for the folder. If a - symbol is displayed, you have replied to the message. If a \* symbol is displayed after the date, the **Date:** field was not present and the displayed date is the last date the message was changed.

### Flags

- clear** Clears the display after sending output. **scan** uses the values of the **\$TERM** and **\$TERMCAP** environment variables to determine how to clear the display. If standard output is not a display, **scan** sends a form feed character after sending the output.
- +folder msgs** Displays information about each specified message in the specified folder. You can use the following message references when specifying *msgs*:
- |             |              |                 |
|-------------|--------------|-----------------|
| <i>num</i>  | <b>first</b> | <b>prev</b>     |
| <b>cur</b>  | .            | <b>next</b>     |
| <b>last</b> | <b>all</b>   | <i>sequence</i> |
- The default folder is the current folder. If a folder is specified, it becomes the current folder. The default for *msgs* is **all**.
- form file** Displays the **scan** command output in the alternate format described by *file*.
- format string** Displays the **scan** command output in the alternate format described by *string*.
- header** Displays a heading that lists the folder name and the current date and time.
- help** Displays help information for the command.
- noclear** Does not clear the terminal after sending output. This is the default.
- noheader** Does not display a heading. This is the default.
- width num** Sets the number of columns in the **scan** command output. The default is the width of the display.

### Profile Entries

- Alternate-Mailboxes:** Specifies your mailboxes.
- Current-Folder:** Sets your default current folder.
- Path:** Specifies your *user-mh-directory*.

## Files

`$HOME/.mh-profile`      The MH user profile.

## Related Information

Other MH commands: “**inc**” on page 518, “**pick**” on page 748, “**show**” on page 942.

The **mh-format** and **mh-profile** files in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

## sccsdiff

---

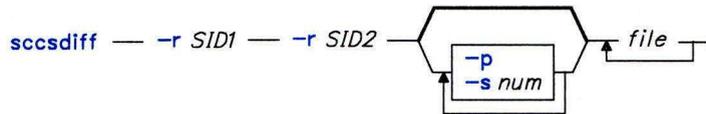
## sccsdiff

---

### Purpose

Compares two versions of a Source Code Control System (SCCS) file.

### Syntax



OL805258

### Description

The **sccsdiff** command reads two versions of an **SCCS** file, compares them, and writes to standard output the differences between the two versions. Any number of **SCCS** files can be specified, but the same arguments apply to all files.

### Flags

- p** Pipes the output through **pr**.
- rSID1** Specifies *SID1* as one delta of the **SCCS** file for **sccsdiff** to compare.
- rSID2** Specifies *SID2* as the other delta of the **SCCS** file for **sccsdiff** to compare.
- snum** Specifies the file segment size for **bdiff** to pass to **diff**. This is useful when **diff** fails due to a high system load.

### Related Information

The following commands: “**bdiff**” on page 102, “**get**” on page 477, “**help**” on page 513, and “**pr**” on page 761.

The **sccsfile** file in *AIX Operating System Technical Reference*.

The discussion of **SCCS** in *AIX Operating System Programming Tools and Interfaces*.

---

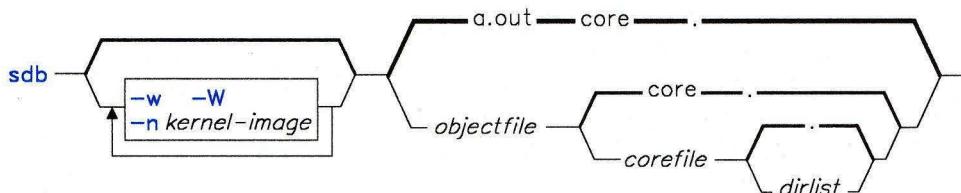
# sdb

---

## Purpose

Provides a symbolic debugger for C and assembler programs.

## Syntax



OL805214

## Description

The **sdb** command provides a symbolic debugger to be used with C and assembler programs. With it you can examine object and core files and provide a controlled environment for running a program. You can set breakpoints at selected statements or run the program one line at a time. You can debug using symbolic variables and instruct **sdb** to display them in their correct format.

Normally, *objectfile* is an executable file produced by invoking **cc** with the **-g** flag. If you have not compiled *objectfile* using the **-g** flag or if it is not executable (because of compiler or loader errors), the symbolic capabilities of **sdb** are limited, but you can still examine the file and debug the program. *objectfile* should always be in the same directory as the source files used to construct it. Its default name is **a.out**.

The *corefile* parameter specifies a core image file. Its default is **core**. The system writes out this core image of a process when it ends abnormally. Specifying **-** (minus) for *corefile* instructs **sdb** to ignore any core image file that may be present. The colon-separated list of directories specified by the *dirlist* parameter identifies the location of the source files used to build *corefile*. The default is the current directory. If *dirlist* is the name of a file, the contents of the file should be a colon-separated list of directory names.

While running, **sdb** always recognizes a **current line** and **current file**. If *corefile* exists, **sdb** initially sets the current line and the current file to the line and the file that contains the source statement at which the process ended. Otherwise, it sets them to the first line in **main** and the file containing **main**. There is also a **current function**, which is the function you are working with at any given time. You can change the current line, file, or function with the **e** command.

Write variable names as you do in C language programs. Access variables local to a function by using the form *function:variable*. The current function is the default function. You can also specify a variable by its address. Since you can use all forms of integer constants which are valid in C, addresses can be expressed as decimal, octal, or hexadecimal values.

Refer to structure members as *variable.member*, pointers to structure members as *variable->member*, and array elements as *variable[number]*.

If you use the form *number.member* or *number->member*, **sdb** assumes *number* to be the address of the last structure referred to. Generally, **sdb** interprets a structure as a set of variables. Thus, it displays the values of all elements when you request it to display a structure. If, however, you request the address of a structure, it displays this value and not the addresses of individual elements.

Refer to elements of a multidimensional array as *variable[number][number]* or as *variable[number,number]*. In place of *number*, use the form *number;number* to indicate a range of values. You can also use an \* (asterisk) to represent all legitimate values for a subscript or omit subscripts to indicate the full range of values. As with structures, **sdb** displays all the values of an array or of a section of an array if you omit trailing subscripts. If you omit subscripts, it displays only the address of the array itself or of the section specified.

Refer to a variable on the stack by using the form *function:variable,*. Here, *number* specifies the variable's location on the stack, counting the top, or most recently pushed variable, as the first. Use this for recursive function calls. The current function is the default.

Refer to line numbers as *filename:number* or *function:number*. The current file and current function are the default values.

### Notes:

1. Data stored in text sections is indistinguishable from functions.
2. Line number information in optimized functions is unreliable, and some information may be missing.
3. Source line and local symbol information for routines in shared libraries is not implemented, and these modules should not be compiled with the **-g** flag. Break points may be set in these routines by address only, and code in shared library modules may be single-stepped by instruction only.
4. The **sdb** command cannot comprehend a module in which C functions (as opposed to declarations and preprocessor definitions) occur in include files.

## Flags

- n** *kernel-image* Specifies the name of the running kernel (or the one running when *corefile* was produced). This enables proper traces back through the floating-point emulation code. **/unix** is the default value.
- w** Allows overwriting of locations in *objectfile*.
- W** Turns off the warnings normally given if source files cannot be found or are newer than *objectfile*.

## Subcommands

### Examining Program Data

- T** Displays the top line of the stack trace.
- t** Displays a stack trace of the program that ended abnormally.
- variable*/[*nlf*] Displays *variable* or *n* memory locations starting at the address of *variable*.  
The *l* parameter selects the number of bytes in one memory location. Your choices are:
  - b** One byte
  - h** Two bytes
  - l** Four bytes.
 The *f* parameter selects the display format. This can be one of the following:
  - a** Displays all bytes from the address of the *variable* to the first null byte.
  - c** Displays a character value.
  - d** Displays a decimal value.
  - f** Displays a 32-bit, single-precision floating-point value.
  - g** Displays a 64-bit, double-precision floating-point value.
  - I** Interprets values as assembler language instructions and displays numerically.
  - i** Interprets values as assembler language instructions and displays them numerically and symbolically.
  - o** Displays an octal value.
  - t** Displays F if *variable* = 0; otherwise, displays T.
  - p** Displays a pointer to a function.
  - s** Treats *variable* as a string pointer and displays characters beginning with the address to which it points and ending at the first null byte reached.

- u** Displays an unsigned decimal value.
- x** Displays a hexadecimal value.

If you do not specify *n*, *l*, or *f*, **sdb** chooses a value appropriate to *variable* type as declared in the source file. Specifying a memory location size works only with formats **c**, **d**, **o**, **u**, and **x**. You can specify the number of memory locations (*n*) to be displayed by the **s** or **a** formats.

For strings that contain 2-byte extended characters, the font shift character is represented by a \ (backslash) followed by lowercase **s** and the font shift number.

For example, `\s1` means that the current byte being displayed is a font shift character. This form of representation for the font shift byte is only available when a count is specified. However, if the first character contained in the address specified by the **a** format is the second byte of a 2-byte extended character, then that byte is displayed without the proper shift affixed to construct the whole 2-byte sequence. (When Japanese Language Support is installed on your system, the font shift symbols are not used. Strings can contain any combination of ASCII, 1-byte kana, and kanji characters.)

The default action for these formats is to display characters until either a null byte is reached or 128 characters have been displayed. The command `./` (dot slash) re-displays the last variable.

You can use the special **sh** characters **\*** (asterisk) and **?** (question mark) in function and variable names, providing a limited form of pattern matching. If you give no function name, global variables and variables local to the current function are matched. If you specify a function name, then only variables local to that function are matched. To match only global variables, use the form `:pattern`.

The **sdb** command recognizes structures, arrays, and pointers so that all of the following commands work:

```
array[2][3]/  
sym.id/  
psym->usage/  
xsym[20].p->usage/
```

*line?*[*lf*]  
*variable?*[*lf*]

Displays the value found in *objectfile* at the address selected by *line* or *variable* (function name), using the specified *length* and *format*. The default format is **i**.

- line* = [*lf*]  
*variable* = [*lf*]  
*number* = [*lf*]
- Displays the address of *variable* or *line* or the value of *number* in the specified length and format. The default is **lx**. *number* = [*lf*] provides a convenient way to convert decimal, octal, and hexadecimal values.
- variable*!*value* Sets *variable* to the given *value*. *value* can be a numeric or character constant or another variable. Expressions that produce more than one value, such as structures, are not allowed as *value*. However, *variable* can be an expression which represents more than one variable, such as an array or structure name.
- Specify a character constant with an initial ' (single quote), for example, 'c. Numbers are treated as integers unless they contain a decimal point or an exponent. In the latter case, they are treated as having the type double. Register values are viewed as integers. If you give an address of a variable, it is treated as the address of a variable of type **int**. C conventions are used in any type conversions that are necessary to perform the indicated assignment.
- x** Displays the machine registers and the current assembler language instruction.
- X** Displays the current assembler language instruction.

## Displaying and Manipulating Source Files

- e** *function*  
**e** *file*  
**e** *dir/*  
**e** *dir file*
- Changes the current function, file, or directory. Specifying only *function* also sets the current file to the one containing the selected function. **sdb** reports the current function, file, or directory for any unspecified parameters.
- /pattern/* Searches forward from the current line for a line containing a string matching *pattern*. The trailing / (slash) can be omitted. See “**ed**” on page 371 for a discussion of pattern notation.
- ?pattern?* Searches backward from the current line for a line containing a string matching *pattern*. The trailing ? (question mark) can be omitted.
- p** Displays the current line.
- z** Displays the current line and the following nine lines. Sets the current line to the last line displayed.
- w** Displays the 10 lines around the current line (a window).
- number* Sets the current line to *number*. Displays the new current line.

- number* + Advances the current line by the specified *number* of lines. Displays the new current line.
- number*- Decreases the current line by the specified *number* of lines. Displays the new current line.
- Ctrl-D** Scrolls. Pressing **Ctrl-D** displays the next 10 lines of source or data.
- Enter** If the previous command displayed a source line, pressing the **Enter** key advances the current line by one line and displays the new current line. If the previous command displayed a memory location, pressing the **Enter** key displays the next memory location.

### Controlling the Running of the Source Program

- [*num*] **r** [*p* [*p2*] . . . ]  
[*num*] **R** Runs the program with the given parameters. If you specify no parameters with **r**, it reuses previously specified parameters. **R** runs the program with no parameters. A parameter beginning with < (left angle bracket) or > (right angle bracket) redirects input or output, respectively. If given, *num* selects the number of breakpoints to be ignored.
- [*line*] **b** [*command*[: *command*] . . . ]  
Sets a breakpoint at the given line. If you specify a function name without a line number, **sdb** places a breakpoint at the first line in the function, even if it was not compiled with the **-g** flag. If you do not specify a *line*, a breakpoint is placed at the current line. If you specify no *commands*, the program stops running just before the breakpoint and returns control to **sdb**. Otherwise **sdb** performs the specified *commands* when the breakpoint is encountered, and then the program being debugged continues. If the **k** command is specified, however, control returns to **sdb**.
- B** Lists the currently active breakpoints.
- [*line*] **d** Deletes a breakpoint at the selected line. If you select no *line*, breakpoints are deleted interactively. **sdb** displays each breakpoint location and reads a line from standard input. If the line begins with a **y** or **d**, then it deletes the breakpoint.
- D** Deletes all breakpoints.
- [*line*] **c** [*num*]  
[*line*] **C** [*num*] Continues running program after a breakpoint or an interrupt. **C** continues after resetting the signal that caused the program to stop. **c** ignores the signal. An optional *num* selects the number of breakpoints to ignore. If you specify a *line*, **sdb** places a temporary breakpoint at the line and continues the program. It deletes the breakpoint when the command finishes.

- [line] g [num]** Continues after a breakpoint, with execution resumed at the given line. *num* specifies how many breakpoints to ignore.
- l** Displays the last executed line.
- i**
- I** Runs the program one machine level instruction at a time, ignoring the signal that stopped the program (**i**) or passing the signal back to the program (**I**).
- s [num]**
- S [num]** Runs the program for one or the specified number of lines. **S** is equivalent to **s** except that it does not stop within called functions. Use **S** if you are confident that the called function works, but want to test the calling routine.
- variable\$m [num]**
- address:m [num]** Runs the program until the specified location is modified with a new value or is modified a specified *num* of times. The *variable* must be accessible from the current function.
- line a** If *line* is of the form *function:number*, this command has the effect of the **sdb** subcommand: *line* b l. If *line* is of the form *function:*, it has the effect of the **sdb** subcommand: *function:* b T.
- [level] v** Toggles verbose mode, for use with the **S**, **s** or **m** commands. If you omit *level*, then just the current source file or subroutine name is displayed when either changes. If *level* is 1 or greater, each C source line is displayed before it is executed; if *level* is 2 or greater, each assembler statement is also displayed. If verbose mode is on for any level, another **v** turns it off.
- function(p [p . . . ])/f** Runs the named function, passing to it the specified parameters. These can be integer, character, or string constants or the names of variables accessible to the current function. You can specify the format of displayed values. The default format is **d** (decimal).
- k** Kills the program being debugged.
- M** Displays the address maps. Program addresses are mapped to file addresses using two field triples: *b1, e1, f1* and *b2, e2, f2*. The *f1* field is the length of the header at the beginning of the file; the *f2* field is the displacement from the beginning of the file to the data. For a plain executable file with mixed text and data, this is the same as the length of the header; for shared text and split instruction/data files, this is the length of the header plus the size of the text portion. The *b* and *e* fields are the starting and ending locations for a segment.

Given an address *A*, calculate its location in the file (either **a.out** or **core**) as follows:

If  $b1 < A < e1$   
then *file address* = (*A-b1*) *f1*

If  $b2 < A < e2$   
then *file address* = (*A-b2*) *f2*

- M**[?/][\*] *b e f* Records new values for the address map. The parameters ? and / specify the text and data maps respectively. The first segment is changed unless you specify \*, in which case the second segment is changed. (These segments differ only in programs with split instruction and data space. In this case, use the second segment to examine the data section of the **a.out** file rather than the data section of the core image file.) If you give fewer than three values, the remaining map parameters are left unchanged.
- " *string* Displays the given string. **sdb** recognizes C escape sequences of the form *\character*.
- !*AIX-command* Performs the command specified by *AIX-command*.
- < *file* Reads commands from *file* until reaching the **end-of-file** character and then continues to accept commands from standard input. When a command in such a file tells **sdb** to display a variable, the variable name is displayed along with the value. This command cannot be nested.
- > *file* Redirects standard output to *file*.
- q** Exits the debugger.

### Debugging the Debugger

- Q** Displays a list of functions and files being debugged.  
**V** Displays the version number.

## Files

a.out, core

## Related Information

The following commands: “**cc**” on page 140, “**ed**” on page 371, and “**sh**” on page 913.

The **a.out** and **core** files in *AIX Operating System Technical Reference*.

“Overview of International Character Support” in *Managing the AIX Operating System*.

“Debugging Programs” in *AIX Operating System Programming Tools and Interfaces*.

The discussion of Japanese Language Support in *Japanese Language Support User’s Guide*.

---

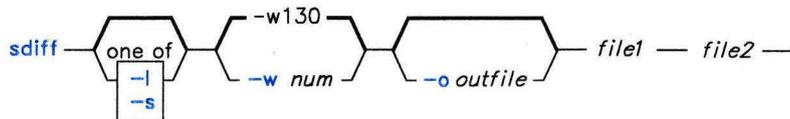
# sdiff

---

## Purpose

Compares two files and displays the differences in a side by side format.

## Syntax



OL805301

## Description

The **sdiff** command reads *file1* and *file2*, uses **diff** to compare them, and writes the results to standard output in a side by side format. **sdiff** displays each line of the two files with a series of blanks between them if the lines are identical, a < (less than sign) in the field of blanks if the line only exists in *file1*, a > (greater than sign) if the line only exists in *file2*, and a | (vertical bar) for lines that are different.

When you specify the **-o** flag, **sdiff** produces a third file by merging *file1* and *file2* according to your instructions.

**Note:** The **sdiff** command invokes **diff -b** to compare two input files. The **-b** flag causes **diff** to ignore trailing spaces, tab characters, and consider other strings of blanks as equal.

## Flags

- l** Displays only the left side when lines are identical.
- o outfile** Creates a third file, *outfile*, by a controlled line-by-line merging of *file1* and *file2*. The following subcommands govern the creation of this file:
  - l** Adds the left side to *outfile*.
  - r** Adds the right side to *outfile*.
  - s** Stops displaying identical lines.

## sdiff

---

- v** Begins displaying identical lines.
- e l**  
**e r**  
**e b**  
**e** Starts **ed** with the left side, the right side, both sides, or an empty file, respectively.
- Each time you exit from **ed**, **sdiff** writes the resulting edited file to the end of *outfile*. If you fail to save the changes before exiting, **sdiff** writes the initial input to *outfile*.
- q** Exits the program.
- s** Does not display identical lines.
- w num** Sets the width of the output line to *num*, 130 characters, by default.

## Examples

1. To print a comparison of two files:

```
sdiff chap1.bak chap1 | print
```

This prints a side by side listing that compares each line of `chap1.bak` and `chap1`. The `| print` sends the listing to the **print** command. **sdiff** assumes that your printer has wide paper (130 columns).

2. To display only the lines that differ:

```
sdiff -s -w 80 chap1.bak chap1
```

This displays the differences at the work station. The **-w 80** sets page width to 80 columns. The **-s** flag tells **sdiff** not to display lines that are identical in both files.

3. To selectively combine parts of two files:

```
sdiff -s -w 80 -o chap1.combo chap1.bak chap1
```

This combines `chap1.bak` and `chap1` into a new file called `chap1.combo`. For each group of differing lines, **sdiff** asks you which group to keep or whether you want to edit them using **ed**.

## Related Information

The following commands: “**diff**” on page 320 and “**ed**” on page 371.

## secure

---

### Purpose

Establishes a more secure system configuration.

### Syntax

```
secure —|
```

OL805480

### Description

The **secure** command increases the trustworthiness of the system. It configures trusted path management, login protection, and proper file system permissions. This command is a shell script and can be altered by an administrator of the system.

The **secure** command should be run after the initial system installation. This command can be run more than once, but it will not reverse any previous configuration set by the **secure** command.

#### Trusted Path Management

For trusted path management, **secure** does the following:

- Adds the `sak = on` attribute to the default stanza of the `/etc/ports` file. This attribute enables the secure attention key (SAK) for the ports defined in `/etc/ports`.
- Adds the `synonym = /dev/hft` attribute to the `/dev/concole` stanza of the `/etc/ports` file. The addition of this attribute ensures that the protection and ownership of all the virtual terminals will match that of the console.
- Adds the `shell = /bin/actman` attribute to the `/dev/concole` stanza of the `/etc/ports` file. This addition will ensure that all the virtual terminals are closed before a user can log off the console.

### Login Protection

To enhance login protection, **secure** does the following:

- Removes any contents from **/etc/autolog**. This action prevents anyone from logging in without proper authentication.
- Prohibits users from logging in as superusers (**root**). Superuser privileges can still be obtained with the **su** command.
- Causes the default shell (or login shell) for **root** to be the trusted shell **/bin/tsh**.

### File System Permissions

For proper file system permissions, **secure** does the following:

- Checks the trusted computing base (TCB) to ensure proper permissions. In addition, **secure** identifies files not in the TCB that might be untrustworthy. This identification is done with the **sysck** command.
- Adds the **sysck** command to the **/etc/rc** file to run this command at system initialization.

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

## Related Information

The following command: “**sysck**” on page 1031.

The **sysck.cfg** and **ports** file formats in *AIX Operating System Technical Reference*.

The discussion of security management in *Managing the AIX Operating System*.

---

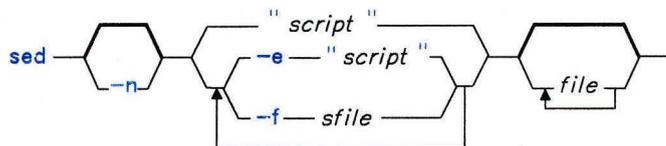
# sed

---

## Purpose

Provides a stream editor.

## Syntax



OL805302

## Description

The **sed** command modifies lines from the specified *file* according to an edit script and writes them to standard output. The **sed** command includes many features for selecting lines to be modified and making changes only to the selected lines.

The **sed** command uses two work spaces for holding the line being modified: the **pattern space**, where the selected line is held, and the **hold space**, where a line can be stored temporarily.

An edit script consists of individual subcommands, each one on a separate line. The general form of **sed** subcommands is:

```
[address-range] function[modifiers]
```

The **sed** command processes each input *file* by reading an input line into a pattern space, applying all **sed** subcommands in sequence whose addresses select that line, and writing the pattern space to standard output. It then clears the pattern space and repeats this process for each line in the input *file*. Some of the subcommands use a hold space to save all or part of the pattern space for subsequent retrieval.

When a command includes an address, either a line number or a search pattern, only the addressed line or lines are affected by the command. Otherwise, the command is applied to all lines.

An address is either a decimal line number, a \$ (dollar sign), which addresses the last line of input, or a context address. A context address is a regular expression similar to those used in **ed** except for the following differences:

- You can select the character delimiter for patterns. The general form of the expression is:

`\?pattern?`

where ? is a character delimiter you select. This delimiter cannot be a 2-byte international character support extended character. The default form for the pattern is:

`/pattern/`

- The sequence `\n` matches a new-line character in the pattern space, except the terminating new line.
- A `.` (dot) matches any character except a terminating new-line character. That is, unlike **ed**, which cannot match a new-line character in the middle of a line, **sed** can match a new-line character in the pattern space.

Certain commands allow you to specify one line or a range of lines to which the command should be applied. These commands are called **addressed commands**. The following rules apply to addressed commands:

- A command line with no address selects every line.
- A command line with one address, expressed in context form, selects each line that matches the address.
- A command line with two addresses separated by commas selects the entire range from the first line that matches the first address through the next line that matches the second. (If the second address is a number less than or equal to the line number first selected, only one line is selected.) Thereafter the process is repeated, looking again for the first address.

### Notes:

1. The *text* parameter accompanying the `a\`, `c\`, and `i\` commands can continue onto more than one line provided all lines but the last end with a `\` to quote the new-line character. Back slashes in text are treated like back slashes in the replacement string of an `s` command and can be used to protect initial blanks and tabs against the stripping that is done on every script line. The *rfile* and *wfile* parameters must end the command line and must be preceded by exactly one blank. Each *wfile* is created before processing begins.
2. The **sed** command can process up to 99 commands in a pattern file.

## Flags

- e "script"** Uses the string *script* as the editing script. If you are using just one **-e** flag and no **-f** flag, the **-e** flag can be omitted.
- f sfile** Uses *sfile* as the source of the edit script. *sfile* is a prepared set of editing commands to be applied to *file*.
- n** Suppresses all information normally written to standard output.

## Subcommands

In the following list of functions, the maximum number of permissible addresses for each function is indicated in parentheses. The **sed** script subcommands are as follows:

- (1) **a\**  
*text* Places *text* on the output before reading the next input line.
- (2) **b[label]** Branches to the **:** command bearing the *label*. If *label* is empty, it branches to the end of the script.
- (2) **c\**  
*text* Deletes the pattern space. With 0 or 1 address or at the end of a 2-address range, places *text* on the output. Starts the next cycle.
- (2) **d** Deletes the pattern space. Starts the next cycle.
- (2) **D** Deletes the initial segment of the pattern space through the first new-line character. Starts the next cycle.
- (2) **g** Replaces the contents of the pattern space by the contents of the hold space.
- (2) **G** Appends the contents of the hold space to the pattern space.
- (2) **h** Replaces the contents of the hold space by the contents of the pattern space.
- (2) **H** Appends the contents of the pattern space to the hold space.
- (1) **i\**  
*text* Writes *text* to standard output before reading the next line into the pattern space.
- (2) **l** Writes the pattern space to standard output showing nondisplayable characters as 2- or 4-digit hexadecimal values. Long lines are folded.
- (2) **n** Writes the pattern space to standard output. Replaces the pattern space with the next line of input.
- (2) **N** Appends the next line of input to the pattern space with an embedded new-line character. (The current line number changes.) You can use this to search for patterns that are split onto two lines.

- (2)**p** Writes the pattern space to standard output.
  - (2)**P** Writes the initial segment of the pattern space through the first new-line character to standard output.
  - (1)**q** Branches to the end of the script. Does not start a new cycle.
  - (2)**r *rfile*** Reads the contents of *rfile*. Places contents on the output before reading the next input line.
  - (2)**s/*pattern*/*replacement*/*flags***  
Substitutes *replacement* string for the first occurrence of the *pattern* in the pattern space. Any character appearing after the *s* can substitute for the / separator.
- 

## Japanese Language Support Information

Any single-byte character appearing after the *s* can substitute for the /.

---

You can add zero or more of the following *flags*:

- g** Substitutes all nonoverlapping instances of the *pattern* rather than just the first one.
  - p** Writes the pattern space to standard out if a replacement was made.
  - w *wfile***  
Writes the pattern space to *wfile* if a replacement was made. Appends the pattern space to *wfile*. If *wfile* was not already created by a previous write by this **sed** script, **sed** creates it.
  - (2)**t*label*** Branches to **:*label*** in the script file if any substitutions were made since the most recent reading of an input line execution of a **t** subcommand. If you do not specify *label*, control transfers to the end of the script.
  - (2)**w*wfile*** Appends the pattern space to *wfile*.
  - (2)**x** Exchanges the contents of the pattern space and the hold space.
  - (2)**y/*pattern1*/*pattern2*/**  
Replaces all occurrences of characters in *pattern1* with the corresponding characters *pattern2*. The byte lengths of *pattern1* and *pattern2* must be equal.
- 

## Japanese Language Support Information

The character lengths of these two patterns must be equal.

---

---

<code>(2)!sed-cmd</code>	Applies the specified <b>sed</b> subcommand only to lines <i>not</i> selected by the address or addresses.
<code>(0):label</code>	This script entry simply marks a branch point to be referenced by the <b>b</b> and <b>t</b> commands. This label can be any sequence of eight or fewer bytes.
<code>(1)=</code>	Writes the current line number to standard output as a line.
<code>(2){subcmd</code>	.
	.
	.
<code>}</code>	Groups subcommands enclosed in {} (braces).

## Examples

1. To perform a global change:

```
sed "s/happy/enchanted/g" chap1 >chap1.new
```

This replaces each occurrence of `happy` found in the file `chap1` with `enchanted`, and puts the edited version in a separate file named `chap1.new`. The `g` at the end of the `s` subcommand tells **sed** to make as many substitutions as possible on each line. Without the `g`, **sed** replaces only the first `happy` on a line.

The **sed** stream editor operates as a filter. It reads text from standard input or from the files named on the command line (`chap1` in this example), modifies this text, and writes it to standard output. Unlike most editors, it does not replace the original file. This makes **sed** a powerful command when used in pipelines.

2. To use **sed** as a filter in a pipeline:

```
pr chap2 | sed "s/Page *[0-9]*$/(&)/" | print
```

This encloses the page numbers in parentheses before printing `chap2`. The `pr` command puts a heading and page number at the top of each page, then **sed** puts the page numbers in parentheses, and the `print` command prints the edited listing.

The **sed** pattern `/Page *[0-9]*$/` matches page numbers that appear at the end of a line. The `s` subcommand changes this to `(&)`, where the `&` stands for the page number that was matched.

3. To display selected lines of a file:

```
sed -n "/food/p" chap3
```

This displays each line in `chap3` that contains the word `food`. Normally, **sed** copies every line to standard output after it is edited. The `-n` flag stops **sed** from doing this. You then use subcommands like `p` to write specific parts of the text. Without the `-n`,

this example would display *all* the lines in chap3, and it would show each line containing food twice.

4. To perform complex editing:

```
sed -f script.sed chap4 >chap4.new
```

It is always a good idea to create a **sed** script file when you want to do anything very complex. You can then test and modify your script before using it. You can also reuse your script to edit other files. Create the script file with an interactive text editor.

5. A sample **sed** script file:

```
:join
/\\$/ {N
s/\\n//
b join
}
```

This **sed** script joins each line that ends with a `\` (backslash) to the line that follows it. First, the pattern `/\\$/` selects a line that ends with a `\` for the group of commands enclosed in `{ }`. The `N` subcommand then appends the next line, imbedding a new-line character. The `s/\\n//` deletes the `\` and imbedded new-line character. Finally, `b join` branches back to the label `:join` to check for a `\` at the end of the newly joined line. Without the branch, **sed** writes the joined line and read the next one before checking for a second `\`.

**Note:** The `N` subcommand causes **sed** to stop immediately if there are no more lines of input (that is, if `N` reads an end-of-file character). It does not copy the pattern space to standard output before stopping. This means that if the last line of the input ends with a `\`, then it is not copied to the output.

## Related Information

The following commands: “**awk**” on page 81, “**ed**” on page 371, and “**grep**” on page 501.

The discussion of Japanese Language Support in *Japanese Language Support User’s Guide*.

---

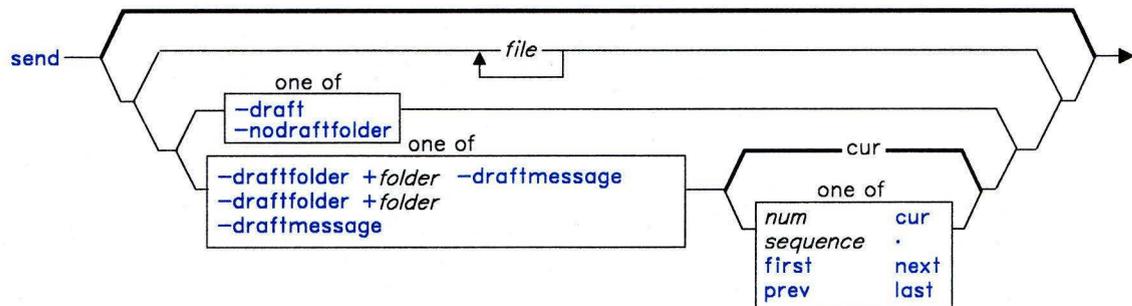
# send

---

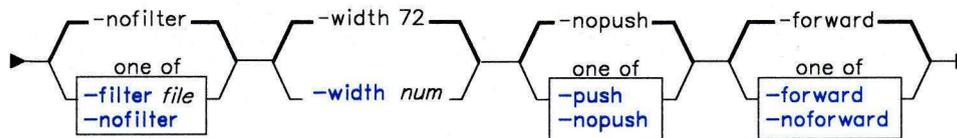
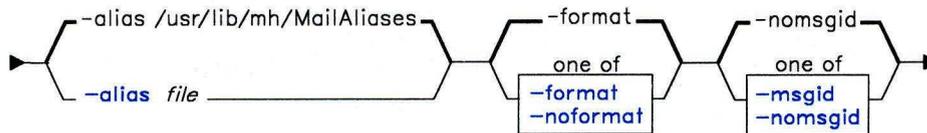
## Purpose

Sends a message.

## Syntax



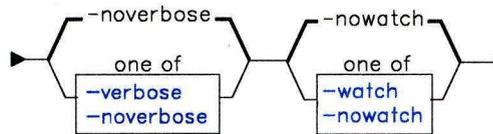
AJ2FL245



AJ2FL246

# send

---



send — -help —|

AJ2FL170

## Description

The **send** command is used to route messages to the mail delivery system by way of the **post** command or the **spost** command. **send** is part of the MH (Message Handling) package and can be used with other MH and AIX commands.

The **send** command causes **From:** and **Date:** fields to be added to each specified message. **send** places the sender's address in the **From:** field unless a **\$SIGNATURE** environment variable is set or a **signature:** profile entry is present. In either of these cases **send** uses the signature. **send** puts the current date in the **Date:** field. If the **dist** command calls **send**, **send** prepends **Resent-** to the **From:**, **Date:**, and **Message-ID:** fields.

The **send** command then takes each of the specified message files and calls the **post** command or the **spost** command to deliver them. After successful delivery, **send** renames the message file by placing a comma in front of the file name. This comma removes the message from the folder without actually deleting the file. The file is retrievable until you send another message. If the delivery fails, **send** displays an error message.

## Flags

- alias** *file* Specifies that *file* is a mail alias file to be searched for aliases. The default alias file is **/usr/lib/mh/MailAliases**.
- draft** Uses the current draft message if no file is specified. Without this flag, **send** asks the user if the current draft message is the one to use when no file is specified.
- draftfolder** + *folder* Specifies the draft folder that contains the draft message to be sent. If **-draftfolder** + *folder* is followed by *msg*, *msg* represents the **-draftmessage** attribute.

- 
- draftmessage** *msg* Specifies the draft message to be sent. You can use one of the following message references as *msg*:
- |             |                 |              |
|-------------|-----------------|--------------|
| <i>num</i>  | <i>sequence</i> | <b>first</b> |
| <b>prev</b> | <b>cur</b>      | .            |
| <b>next</b> | <b>last</b>     | <b>new</b>   |
- The default draft message is **cur**.
- filter** *file* Uses the format instructions in the specified file to reformat copies of the message sent to **Bcc:** recipients.
- format** Puts all recipient addresses in a standard format for the delivery transport system. This flag is the default.
- forward** Adds a failure message to the draft message and returns it to the sender if the **send** command fails to deliver the draft. This flag is the default.
- help** Displays help information for the command.
- msgid** Adds a message-identification component (such as **Message-ID:**) to the message.
- nodraftfolder** Undoes the last occurrence of **-draftfolder** + *folder*. This flag is the default.
- nofilter** Removes the **Bcc:** header from the message for recipients listed in the **To:** and **cc:** fields, and sends the message with minimal headers to recipients listed in the **Bcc:** field. This flag is the default.
- noformat** Does not alter the format of the recipient addresses.
- noforward** Does not return the draft message to the sender if delivery fails.
- nomsgid** Does not add a message-identification component. This flag is the default.
- nopush** Runs the **send** command in the foreground (see the **-push** flag). This flag is the default.
- noverbose** Does not display information during the delivery of the message to the **sendmail** command. This flag is the default.
- nowatch** Does not display information during delivery by the **sendmail** command. This flag is the default.
- push** Runs the **send** command in the background. **send** does not display error messages on the terminal if delivery fails. You can use the **-forward** flag to return messages to you that fail on delivery.

## send

---

<b>-verbose</b>	Displays information during the delivery of the message to the <b>sendmail</b> command. This information allows you to monitor the steps involved.
<b>-watch</b>	Displays information during the delivery of the message by the <b>sendmail</b> command. This information allows you to monitor the steps involved.
<b>-width <i>num</i></b>	Sets the width of components that contain addresses. The default is 72 columns.

## Profile Entries

<b>Draft-Folder:</b>	Sets your default folder for drafts.
<b>mailproc:</b>	Specifies the program used to post failure notices.
<b>Path:</b>	Specifies your <i>user-mh-directory</i> .
<b>postproc:</b>	Specifies the program used to post messages.
<b>Signature:</b>	Sets your mail signature.

## Files

`$HOME/.mh-profile`      The MH user profile.

## Related Information

The following commands: “**ali**” on page 48, “**comp**” on page 185, “**dist**” on page 336, “**forw**” on page 438, “**post**” on page 758, and “**sendmail**” on page 897.

The **mh-alias**, **mh-format**, and **mh-profile** files in *AIX Operating System Technical Reference*.

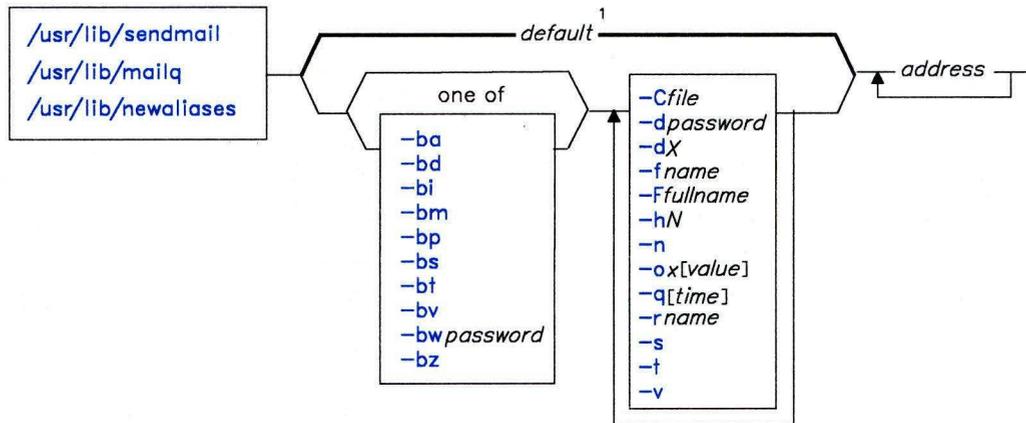
“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

# sendmail

## Purpose

Routes mail for local or network delivery.

## Syntax



<sup>1</sup>default = -bm for **sendmail**  
 -bp for **mailq**  
 -bi for **newaliases**

`/usr/lib/mailq` —|

`/usr/lib/newaliases` —|

AJ2FL126

AJ2FL145

AJ2FL147

## Description

The **sendmail** program receives formatted text messages and routes the message to one or more other users on the local system, or if connected to a network, to users on other systems. The program translates the format of message heading information to match the requirements of the destination system. It determines which network to use based on the syntax and content of the addresses.

## sendmail

---

The program can deliver messages to:

- Users on the local system
- Users connected to the local system using the TCP/IP protocol
- Users connected to the local system using the **uucp** protocol.

The **sendmail** program operates mainly as a background, mail-routing program. Other programs, such as **Mail** and the message handler routines, provide user interfaces for generating and receiving mail that **sendmail** handles. However, if you enter the **sendmail** command with no flags, it reads standard input for the message text until it receives a **Ctrl-D** or a line with only a single period, designating the end of the message. Then it sends a copy of the message to all addresses listed. For example, the following input at the command line sends the message This is a test message to the mailbox for user george on the local system:

```
$ /usr/lib/sendmail george
This is a test message
.
$
```

The **sendmail** program uses a configuration file (**/usr/adm/sendmail/sendmail.cf** by default) to set many operational parameters and to determine how the program parses addresses. This file is a text file that you can edit. However, **sendmail** uses a compiled form of this file (**/usr/adm/sendmail/sendmail.cfDB**). For any changes made to this file to be effective, you must build a new copy of the compiled configuration file by running **sendmail** with the **-bz** flag.

The **sendmail** program also allows you to define aliases to use when addressing mail handled by the local **sendmail** program. **Aliases** are alternate names that can be used in place of elaborate network addresses. You can also use aliases to build distribution lists. Define aliases in **/usr/adm/sendmail/aliases**. This file is a text file you can edit. However, **sendmail** uses a database version of this file that is kept in the directory **/usr/adm/sendmail/aliasesDB**. For any changes made to the aliases file to be effective, you must build a new alias database by running **sendmail** with the **-bi** flag. If the **sendmail** daemon is running, you must also stop that process and start the daemon again before it recognizes the new alias data base file. Normally the sender of a message is not included when **sendmail** expands an alias address. For example, if amy sends a message to alias D998 and she is defined as a member of that alias, **sendmail** does not send a copy of the message to amy.

Every system must have a user or user alias designated **postmaster**. Assign this alias in the file **/usr/adm/sendmail/aliases**. Unless you change the entry in this file, this alias is assigned to **root**. This alias allows other users outside your system to send mail to a known ID (i.e. **postmaster**) to get information about mailing to users on your system. Also, users on your system can send problem notifications to this ID.

Two additional commands are links to **sendmail**:

<b>/usr/lib/mailq</b>	Prints the contents of the mail queue. This command is the same as running <b>sendmail</b> with the <b>-bp</b> flag.
<b>/usr/lib/newaliases</b>	Builds a new copy of the alias data base from the file <b>/usr/adm/sendmail/aliases</b> . This command is the same as running <b>sendmail</b> with the <b>-bi</b> flag.

## Mail Addresses

Mail addresses are based on the domain address (Arpanet) protocol. These addresses have the following form:

*user@host.domain*

**Note:** The configuration file provided with **sendmail** specifies that blanks in addresses be converted to periods before being transmitted. This convention follows the Arpanet mail protocol as described in RFC822, but does not match the Arpanet mail protocol as described in RFC733 (NIC41952). You can change this setting by setting the **OB** option in the **sendmail** configuration file.

A **domain** is a logical grouping of systems that are connected together by physical network links. No direct relationship exists between the actual physical interconnections and the way in which the systems are grouped in the domain. The **domain name** identifies a specific domain within a larger group of domains. The domain name has the format of a tree structure. Each node (or leaf) on the tree corresponds to a resource set, and each node can create and contain new domains below it. The actual domain name of a node is the path from the root of the tree to that node. Domain names do not correspond to system names, host addresses, or any other type of information.

For example, if node her a is part of the domain IBM, which is in turn a subdomain of COM, and a message is sent to geo at that address it is sent to:

[geo@hera.IBM.COM](mailto:geo@hera.IBM.COM)

The message router (usually **sendmail**) must determine how to send the message to its final destination. If the router is at her a, it delivers the message to user geo. If the router is at another system within the IBM domain, it corresponds with the name server for that domain to find out how to deliver the message. If the router is not a part of the IBM domain, but is in a domain that is under the COM domain, it corresponds with the name server for the COM domain to find out how to deliver the message. The respective name server returns a network address to the router. That network address determines the actual path that the message takes to its destination.

## sendmail

---

The domain address is read from right to left, with each domain in the address separated from the next domain with a . (period). This format does not imply any routing. Thus, although the example is specified as a COM address, the message might actually travel by a different route if that were more convenient or efficient. At some sites, the message associated with the sample address might go directly from the sender to node *hera* over a local area network. At other sites it might be sent over a **uucp** network or a combination of other delivery methods.

Normally, the actual routing of a message is handled automatically. However, you can route the message manually through several specified hosts to get it to its final destination. An address using intermediate hosts, called a **route address**, has the following form:

*@hosta,@hostb:user@hostc*

This address specifies that the message should go first to the remote system represented by *hosta*, then to the remote system represented by *hostb*, and finally to the remote system represented by *hostc*. This path is forced even if there is a more efficient route to *hostc*.

In some cases the user can abbreviate the address rather than typing the entire domain name. In general, systems in the same domain do not need to use the full domain name. For example, a user on node *zeus*.IBM.COM can send a message to *geo@hera*.IBM.COM by typing only *geo@hera*, because they are in the same local domain, IBM.COM.

Other mail address formats exist that are used by other mail routing programs (for example, **uucp**). The mail routing program (**sendmail**) converts most of these other formats to a format that the network routing system can use. However, if you use the domain address format, the routing program operates more efficiently.

For example, if **sendmail** receives an address in the following format:

*@host:user*

It converts it to the corresponding domain address format:

*user@host*

Similarly, if **sendmail** receives an address in the following format:

*host!user*

The mail routing program routes the message directly to the **uucp** command (part of the Basic Networking Utilities (BNU)) However, when sending mail via **uucp**, you must include a route address that indicates which BNU host(s) to send the message through to get to the final destination.

To route messages through the BNU network, use one of the following domain address formats. Your choice depends on the way in which the systems at your site are connected:

1. *@system\_name.domain\_name:uucp-route!user-ID*

For example, the address:

*@zeus:hera!amy*

sends a message to user amy on **uucp** host hera by way of system zeus. The address:

*@apollo.802:merlin!lgh*

sends a message to user lgh on **uucp** host merlin via system apollo under the local domain 802.

2. *uucp-route!user-ID@system\_name.domain\_name*

In this case, the address:

*merlin!arthur!amy@hera.802*

sends a message to user amy on system hera under domain 802 via the BNU link merlin through arthur.

3. *system\_name.domain\_name:uucp-route!user-ID@system\_name.domain\_name*

In this example, the address:

*@apollo.802:merlin!arthur!amy@hera.802*

sends a message to user amy on system hera under domain 802 that first goes through apollo, the gateway node for domain 802, and then through the BNU link merlin through arthur. (Including 802 in this example is optional, since the two domain names are identical.)

4. *hosta!hostb!hostc!user*

This example is a purely **uucp** route address:

*zeus!hera!kronos!amy*

sends a message to amy on kronos, via the BNU link zeus through hera.

5. *@hosta.UUCP:@hostb.UUCP:user@hostc*

This example, like the previous one, is a purely **uucp** route address:

*@zeus.UUCP:@hera.UUCP:amy@kronos.UUCP*

sends a message to amy on kronos, via the BNU link zeus through hera.

# sendmail

---

## Return Codes

The **sendmail** program returns an exit status code. These exit codes are defined in `/usr/include/bsd/sysexits.h`. The following table summarizes the meanings of these return codes:

Return Code	Meaning
<b>EX_CANTCREAT</b>	The <b>sendmail</b> program cannot create a file that the user specified.
<b>EX_DATAERR</b>	The user's input data was not correct.
<b>EX_IOERR</b>	An error occurred during I/O.
<b>EX_NOHOST</b>	The <b>sendmail</b> program could not recognize the specified host name.
<b>EX_NOINPUT</b>	The <b>sendmail</b> program either could not find, or could not read, the specified input file.
<b>EX_NOPERM</b>	The user does not have the needed permissions to perform the requested operation.
<b>EX_NOUSER</b>	The <b>sendmail</b> program could not recognize a specified user ID.
<b>EX_OK</b>	The <b>sendmail</b> program successfully completed the operation for all addresses.
<b>EX_OSERR</b>	A temporary operating system error occurred. An example of such an error is cannot fork because too many processes are currently running.
<b>EX_OSFILE</b>	A system file error occurred. For example, a system file (such as <code>/etc/passwd</code> ) does not exist, cannot be opened or has another type of error preventing it from being used.
<b>EX_PROTOCOL</b>	The remote system returned something that was incorrect during a protocol exchange.
<b>EX_SOFTWARE</b>	An internal software error occurred (including bad arguments).

Return Code	Meaning
<b>EX_TEMPFAIL</b>	The <b>sendmail</b> program could not create a connection. Try the request again later.
<b>EX_UNAVAILABLE</b>	A service or resource that <b>sendmail</b> needed was not available.
<b>EX_USAGE</b>	The command syntax was not correct.

## Flags

<b>-ba</b>	Starts <b>sendmail</b> in Arpanet mode. All input lines to the program must end with a carriage return and a line feed ( <b>CR-LF</b> ). The program generates messages with a <b>CR-LF</b> at the end. The program looks at the from and the sender fields to find the name of the sender.
<b>-bd</b>	Starts <b>sendmail</b> to run in the background as a TCP/IP mail routing daemon.
<b>-bi</b>	Builds the alias database files from information defined in <b>/usr/adm/sendmail/aliases</b> . Running <b>sendmail</b> with this flag is the same as running the command, <b>/usr/lib/newaliases</b> .
<b>-bm</b>	Delivers mail in the usual way. This is the default.
<b>-bp</b>	Prints a listing of the mail queue. Running <b>sendmail</b> with this flag is the same as running the command, <b>/usr/lib/mailq</b> .
<b>-bs</b>	Uses the simple mail transfer protocol (SMTP) as described in RFC821. This flag implies all of the operations of the <b>-ba</b> flag that are compatible with SMTP.
<b>-bt</b>	Starts <b>sendmail</b> in address test mode. This mode allows you to enter addresses interactively and watch as <b>sendmail</b> displays the steps it takes to parse the address. Use this mode for debugging the address parsing rules in a new configuration file.
<b>-bv</b>	Starts <b>sendmail</b> with a request to verify the user IDs provided in the <i>address</i> field of the command. The program responds with a message telling which IDs can be resolved to a mailer program. It does not try to collect or deliver a message. Use this mode to validate the format of user IDs, aliases, or mailing lists.
<b>-bz</b>	Builds the compiled version of the configuration file from information in <b>/usr/adm/sendmail/sendmail.cf</b> .
<b>-Cfile</b>	Starts <b>sendmail</b> using an alternate configuration file specified by the <i>file</i> parameter. Use this flag together with <b>-bt</b> to test a new configuration file before installing it as the running configuration file.

## sendmail

---

- dX** Sets debugging value to *X*.
- F*fullname*** Sets the full name of the sender to be the string provided in the *fullname* parameter.
- f*name*** Sets the name of the *from* person (the sender of the mail). This flag can be used only by those administrative user IDs designated in the configuration file with the **T** control line, or if your ID is the ID supplied in *name*.
- hN** Sets the hop count to *N*. The **hop count** is the number of times that the message has been processed by a **sendmail** program (not just the local copy of **sendmail**). The program increments the hop count every time the message is processed. When it reaches a limit, the message is returned with an error message usually caused by alias looping.
- n** Does not do aliasing.
- o*x*[*value*]** Sets option *x*. If the option is a valued option, you must also specify *value*. See Figure 10 on page 905 for possible options, values, and their meanings.
- q[*time*]** Processes saved messages in the queue at the intervals specified by *time*. If *time* is not specified, this flag processes the queue at once. You can specify *time* as a tagged number using the tag *s* for seconds, *m* for minutes, *h* for hours, *d* for days, and *w* for weeks. If the tag letter is omitted and just a number is specified, **sendmail** uses days as the unit of time. For example, **-q2m** processes the queue every two minutes, but **-q2** processes the queue every two days.
- r*name*** An alternate and obsolete form of the **-f** flag.
- t** Reads the **To:**, **Cc:**, and **Bcc:** lines of the message header to determine where to send the message; deletes the **Bcc:** line before transmitting the message; and deletes any addresses in the argument list of the **sendmail** command.
- v** Starts **sendmail** in verbose mode. The program displays messages regarding the status of transmission, the expansion of aliases and any errors that may occur during the sending of the message.

You can also set or remove **sendmail** processing options. Normally, the person responsible for the mail system uses these options. To set these options, use the **-o** flag on the command line or the **O** control line in the configuration file (**sendmail.cf**).

Option	Function
<i>Afile</i>	Uses the named <i>file</i> as an alternate alias file.
<b>Bc</b>	Sets the blank substitution character to the character specified in the parameter <i>c</i> . The <b>sendmail</b> program replaces unquoted spaces in addresses with this character. The supplied configuration file uses the period (.) for this character.
<b>c</b>	If an outgoing mailer program is specified in the configuration file as expensive to use, this option causes <b>sendmail</b> to queue messages for that mailer program without sending them. The queue can be run later when costs are lowest or when the queue is large enough to send the messages efficiently.
<b>dx</b>	Sets the delivery mode to <i>x</i> . Delivery modes are <b>i</b> for interactive (synchronous) delivery, <b>b</b> for background (asynchronous) delivery, and <b>q</b> for queue only (next time the queue is run) delivery. The supplied configuration file uses a value of <b>b</b> .
<b>ex</b>	Sets error processing to mode <i>x</i> . Valid modes are: <ul style="list-style-type: none"> <li><b>m</b> Mails the error message to the user's mailbox.</li> <li><b>w</b> Writes the error message to the terminal or mails it if the user is not logged in.</li> <li><b>p</b> Displays the error message on the terminal (default).</li> <li><b>q</b> Throws away error message and returns the exit status only.</li> <li><b>e</b> Mails the error message to the user's mailbox, but always exits with a zero exit status (normal return).</li> </ul> <p>If the text of the message is not mailed by modes <b>m</b> or <b>w</b> and if the sender is a local user, a copy of the message is appended to the file <b>dead.letter</b> in the sender's home directory.</p>
<b>f</b>	Saves From lines at the front of messages. These lines are normally discarded.
<b>gN</b>	Sets the default group ID to use when calling mailers to the value specified by <i>N</i> .
<b>Hfile</b>	Specifies the name of the SMTP help file ( <b>/usr/adm/sendmail/sendmail.hf</b> by default).

Figure 10 (Part 1 of 3). Configuration Options

## sendmail

---

Option	Function																						
<b>i</b>	Does not interpret a period (.) on a line by itself as a message terminator.																						
<b>Ic</b>	Interpret the special character specified by <i>c</i> as a space character and all real spaces as delimiters when processing address parsing rules. Use this control line if using an editor that saves tabs as spaces (such as <b>INed</b> ) to edit the configuration file. The default setting of this line is I_.																						
<b>Ln</b>	Specifies the log level to be the value supplied in the <i>n</i> parameter. Valid levels and the activities that they represent are (each number includes the activities of all numbers of lesser value and adds the activity that it represents): <table><thead><tr><th>Level</th><th>Activity Logged</th></tr></thead><tbody><tr><td><b>0</b></td><td>Prevents logging.</td></tr><tr><td><b>1</b></td><td>Logs major problems only.</td></tr><tr><td><b>2</b></td><td>Logs message collections and failed deliveries.</td></tr><tr><td><b>3</b></td><td>Logs successful deliveries.</td></tr><tr><td><b>4</b></td><td>Logs messages deferred (for example, because the host is down).</td></tr><tr><td><b>5</b></td><td>Logs messages that are placed in the queue (normal event).</td></tr><tr><td><b>6</b></td><td>Logs unusual but benign incidents (for example, trying to process a locked file).</td></tr><tr><td><b>9</b></td><td>Logs internal queue ID to external message ID mappings. This can be useful for tracing a message as it travels between several hosts.</td></tr><tr><td><b>12</b></td><td>Logs messages that are of interest when debugging.</td></tr><tr><td><b>16</b></td><td>Logs verbose information regarding the queue.</td></tr></tbody></table>	Level	Activity Logged	<b>0</b>	Prevents logging.	<b>1</b>	Logs major problems only.	<b>2</b>	Logs message collections and failed deliveries.	<b>3</b>	Logs successful deliveries.	<b>4</b>	Logs messages deferred (for example, because the host is down).	<b>5</b>	Logs messages that are placed in the queue (normal event).	<b>6</b>	Logs unusual but benign incidents (for example, trying to process a locked file).	<b>9</b>	Logs internal queue ID to external message ID mappings. This can be useful for tracing a message as it travels between several hosts.	<b>12</b>	Logs messages that are of interest when debugging.	<b>16</b>	Logs verbose information regarding the queue.
Level	Activity Logged																						
<b>0</b>	Prevents logging.																						
<b>1</b>	Logs major problems only.																						
<b>2</b>	Logs message collections and failed deliveries.																						
<b>3</b>	Logs successful deliveries.																						
<b>4</b>	Logs messages deferred (for example, because the host is down).																						
<b>5</b>	Logs messages that are placed in the queue (normal event).																						
<b>6</b>	Logs unusual but benign incidents (for example, trying to process a locked file).																						
<b>9</b>	Logs internal queue ID to external message ID mappings. This can be useful for tracing a message as it travels between several hosts.																						
<b>12</b>	Logs messages that are of interest when debugging.																						
<b>16</b>	Logs verbose information regarding the queue.																						
<b>Mx value</b>	Sets the macro <i>x</i> to <i>value</i> . Use this option from the command line only (with the <b>-o</b> flag).																						
<b>m</b>	Sends to the sender (me) also, if the sender is in an alias expansion. Normally, the sender does not receive a copy of the message.																						
<b>Nnetname</b>	Sets the name of the host network to <i>netname</i> . The <b>sendmail</b> program compares the argument of an SMTP <b>HELO</b> command to <i>hostname.netname</i> (it gets the value of <i>hostname</i> from the kernel). If these values do not match, it adds the <i>hostname.netname</i> string to the Received: lines in the message so that messages can be traced accurately.																						

Figure 10 (Part 2 of 3). Configuration Options

Option	Function
<b>o</b>	This option indicates that this message may have old style headers. Without this option, the message has new style headers (commas instead of spaces between addresses). If this option is set, an adaptive algorithm correctly determines the header format in most cases.
<b>Qdir</b>	Sets the directory in which to queue messages to the directory specified by the <i>dir</i> parameter. That directory must exist.
<b>rtime</b>	Sets the timeout for reads from a mailer program to the value specified by <i>time</i> . If no timeout value is set, <b>sendmail</b> waits indefinitely for a mailer to respond. The default value for this timeout is 5 minutes.
<b>Sfile</b>	Sets the mail statistics file to the <i>file</i> . If this file exists, <b>sendmail</b> stores statistics about mail traffic in a database format in this file. Use the <b>mailstats</b> command to read the information in this file. If the indicated file does not exist, no statistic information is saved.
<b>s</b>	Interactive mode delivers mail without going through the mail queue. When this option is specified, mail is passed through the mail queue in interactive mode also. This action ensures that the message being sent is not lost if a delivery problem occurs.
<b>Ttime</b>	Sets the timeout on messages in the queue to the specified <i>time</i> . After a message has been in the queue for this amount of time, <b>sendmail</b> returns the message to the sender. In <b>sendmail.cf</b> that is provided with <b>sendmail</b> , this value is set to three days.
<b>uN</b>	Sets the default user ID to use when calling mailers to the value specified by <i>N</i> .
<b>v</b>	Run in verbose mode.
<b>Y</b>	When this option is specified, <b>sendmail</b> delivers each message in the mail queue from a separate process. This option uses less memory to process the mail queue. Use of this option is not recommended.

Figure 10 (Part 3 of 3). Configuration Options

## Files

/usr/lib/sendmail	Contains the <b>sendmail</b> program.
/usr/lib/mailq	Displays list of the mail queue.
/usr/lib/newaliases	Builds alias database.

## sendmail

---

/usr/lib/mailstats	Displays <b>sendmail</b> statistics found in <b>/usr/adm/sendmail/sendmail.st</b> .
/usr/lib/sendmail.sh	Contains a shell script replacement for <b>sendmail</b> when <b>sendmail</b> is not installed. Only local mail can be delivered with this shell script acting as <b>sendmail</b> .
/usr/adm/sendmail/aliases	Contains the text version of <b>sendmail</b> aliases.
/usr/adm/sendmail/aliasesDB/DB.dir	Contains one of the alias database files.
/usr/adm/sendmail/aliasesDB/DB.pg	Contains one of the alias database files.
/usr/adm/sendmail/aliasesDBl	Contains the alias database lock file.
/usr/adm/sendmail/sendmail.hf	Contains the SMTP help file.
/usr/adm/sendmail/sendmail.cf	Contains the text version of the <b>sendmail</b> configuration file.
/usr/adm/sendmail/sendmail.cfDB	Contains the <b>sendmail</b> configuration database file.
/usr/adm/sendmail/sendmail.cfDBl	Contains the <b>sendmail</b> configuration database lock file.
/usr/adm/sendmail/sendmail.st	Contains the <b>sendmail</b> statistics file.
/usr/adm/sendmail/smdemon.cleanu	Maintains aging copies of the <b>log</b> file in <b>/usr/spool/mqueue</b> .
/etc/rc.sendmail	Contains the shell script to start the <b>sendmail</b> daemon.
/usr/spool/mqueue	Contains the <b>log</b> file and temporary files associated with the messages in the mail queue (the mail queue directory). Temporary files have names that include the mail queue ID ( <i>mqid</i> ) of the message for which the file was created: <i>dfmqid</i> Data file <i>lfmqid</i> Lock file <i>nfmqid</i> Backup file <i>qfmqid</i> Queue control file <i>tfmqid</i> Temporary control file <i>xfmqid</i> Transcript file for session.
/usr/spool/cron/crontabs/root	Contains a commented entry to run <b>sendmail</b> periodically for use when not routing mail to a network. Uncomment that entry to process the mail queue at the interval specified in that <b>cron</b> file.
/bin/uux	Contains the mailer program to deliver <b>uucp</b> mail.
/bin/bellmail	Contains the mailer program to deliver local mail.

## Related Information

The following commands: “**bellmail**” on page 104, “**mail, Mail**” on page 608, “**uux**” on page 1166.

The book *Interface Program for use with TCP/IP*.

The chapter about managing the mail system in *IBM RT Managing the AIX Operating System*.

The chapter about sending and receiving mail in *IBM RT Using the AIX Operating System*.

The file **sendmail.cf** in *AIX Operating System Technical Reference*.

## setdma

---

## setdma

---

### Purpose

Sets the DMA channel of the specified adapter.

### Syntax

setdma one of  
eesdi 0  
eesdi 1

OL805466

### Description

The **setdma** command sets the DMA channel of the specified adapter. The **eesdi 0** sets the DMA adapter channel to 0. The **eesdi 1** sets the DMA adapter channel to 1.

### Related Information

*Installing and Customizing the AIX Operating System.*

---

# setmnt

---

## Purpose

Creates mount table.

## Syntax

```
setmnt —
```

OL805062

## Description

The **setmnt** command reads lines from standard input and writes the **/etc/mnttab** table to standard output (see the **mnttab** file in *AIX Operating System Technical Reference*). The **/etc/mnttab** is needed for both the **mount** and **unmount** commands. **setmnt** creates a **mnttab** entry for each line read. Input lines have the format:

```
filesystem directory
```

where *filesystem* is the name of the file system's special file and *directory* is the root name of that file system. Thus, *filesystem* and *directory* become the first two strings in the **mnttab** entry. *filesystem* and *directory* must not be longer than 100 characters.

The **setmnt** command enforces an upper limit on the maximum number of **mnttab** entries.

## Example

To set the mount table after it has been destroyed or made invalid:

```
setmnt <<end  
hd1 /u  
fd0 /a  
fd1 /b  
end
```

This command sets the mount table to show **/dev/hd1** mounted on **/u**, **/dev/fd0** on **/a**, and **/dev/fd1** on **/b**.

The **<<end** and **end** define a Here Document, which uses the text entered before the **end** line as the standard input for the **setmnt** command. For more details, see "Inline Input Documents" on page 928.

## setmnt

---

### Files

/etc/mnttab

### Related Information

The **mnttab** file in *AIX Operating System Technical Reference*.

---

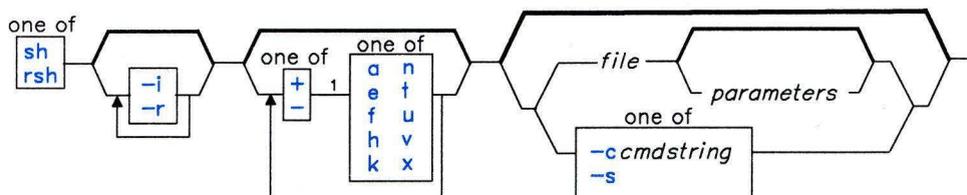
# sh

---

## Purpose

Interprets commands read from a file or entered at the keyboard.

## Syntax



OL805425

<sup>1</sup> Do not put a blank between these items.

OL805308

## Description

The **sh** command is a system command interpreter and programming language. It is not a part of the operating system *kernel*, but an ordinary user program that reads commands entered at the keyboard and arranges for their execution. In addition, it can read commands that you have saved in a file. Such a file is usually called a *shell procedure* or a *command file*. For a complete description of how to write shell procedures to take advantage of this useful tool, see *Using the AIX Operating System*.

A restricted version of shell (the **Rsh** command) is available in the `/bin/Rsh` file that allows you to create user environments with a limited set of privileges and capabilities. See “Restricted Shell” on page 935 for additional information on the restricted shell.

## Commands

A *command* is either a simple command or a *control command* (see “Control Commands” on page 930).

A *simple command* is a sequence of *words* separated by blanks or tabs. A word is a sequence of characters and/or numerals that contains no unquoted blanks. The first word in the sequence (numbered as 0), usually specifies the name of a command. Any remaining words, with a few exceptions, are passed to that command.

The *value* of a simple command is its exit value if it ends normally or (octal) 200 *status* if it ends abnormally. For a list of *status* values, see the **signal** system call in *AIX Operating System Technical Reference*.

A **pipeline** is a sequence of one or more commands separated by a | (vertical bar) or, for historical compatibility, by a ^ (circumflex). In a pipeline, the standard output of each command becomes the standard input of the next command. Each command runs as a separate process, and the shell waits for the last command to end. A **filter** is a command that reads its standard input, transforms it in some way, then writes it to its standard output. A pipeline normally consists of a series of filters. Although the processes in a pipeline (except the first process) can execute in parallel, they are synchronized to the extent that each program needs to read the output of its predecessor.

The exit value of a pipeline is the exit value of the last command.

A **list** is a sequence of one or more pipelines separated by ; (semicolon), & (ampersand), && (two ampersands), or || (two vertical bars) and optionally ended by a ; (semicolon) or an & (ampersand). These separators and terminators have the following effects:

- ;  
Causes **sequential execution** of the preceding pipeline (the shell waits for the pipeline to finish).
  - &  
Causes **asynchronous execution** of the preceding pipeline the shell does *not* wait for the pipeline to finish).
  - &&  
Causes the list following it to be executed *only* if the preceding pipeline returns a zero exit value.
  - ||  
Causes the list following it to be executed *only* if the preceding pipeline returns a nonzero exit value.
- Note:** The **cd** command is an exception. If it returns a nonzero exit value, no subsequent commands in a list are executed, regardless of the separator characters.

The ; and & separators have equal precedence, as do && and ||. The single-character separators have lower precedence than the double-character separators. An unquoted new-line character following a pipeline functions the same as a ; (semicolon).

The shell treats as a comment any word that begins with a # character and ignores that word and all characters following up to the next new-line character.

## Command Execution

Each time the shell executes a command, it carries out the substitutions discussed in the following text. If the command name matches one of the built-in commands discussed in “Built-in Commands” on page 931, it executes it in the shell process. If the command name does not match a built-in command but matches the name of a defined function, it executes the function in the shell process. The shell sets the positional parameters to the parameters of the function.

---

If the command name matches neither a built-in command nor the name of a defined function and the command names an executable file that is a compiled (binary) program, the shell (as *parent*) spawns a new (*child*) process that immediately runs the program. If the file is marked executable but is not a compiled program, the shell assumes that it is a shell procedure. In this case, the shell spawns another instance of itself (a *subshell*), to read the file and execute the commands included in it (note how this differs from the execution of functions). The shell also executes a parenthesized command in a subshell (see page 931). From your point of view as a user, a compiled program is run in exactly the same way as a shell procedure.

The shell normally searches for commands in four places in the file system. The shell first looks for the command in the `/bin` directory. If it does not find the command there, it looks in the `/usr/bin` directory. If this also fails, it looks in the `/etc` directory and then, finally, in the current directory. You can also give a specific path name when you invoke a command, for example `/bin/sort`, in which case the shell does not search any directories other than the one you specify in the path name. If the command name contains a `/` (slash), the shell does not use the search path (note that the restricted shell will not execute such commands). You can give a full path name that begins with the root directory (as in `/bin/sort`), or a path name relative to the current directory, for example `bin/myfile`. In this last case, the shell looks in the current directory for a directory named `bin` and in that directory for `myfile`.

You can change the particular sequence of directories searched by resetting the **PATH** variable (page 923).

The shell remembers the location in the search path of each executed command (to avoid unnecessary **execs** later). If the command was found in a *relative directory* (one whose name does not begin with `/`), the shell must redetermine its location whenever the current directory changes. The shell forgets all remembered locations whenever you change the **PATH** variable or execute the **hash -r** command (page 932).

## Signals

The shell ignores **INTERRUPT** and **QUIT** signals for an invoked command if the command is terminated with a `&` (that is, if it is running in the background). Otherwise signals have the values inherited by the shell from its parent, with the exception of signal 11 (see also the built-in **trap** command on page 934).

## The .profile File

When you log in, the shell is called to read your commands. Before it does that, however, it checks to see if a file named `/etc/profile` exists on the system, and if it does, it reads commands from it (this file should set variables needed by all users). After this, the shell looks for a file named `.profile` in your login directory. If it finds one, it executes commands from it. Finally, the shell is ready to read commands from your standard input.

## File-name Substitution

Command parameters are very often file names. You can automatically produce a list of file names as parameters on a command line by specifying a pattern that the shell matches against the file names in a directory.

Most characters in such a pattern match themselves, but you can also use some special pattern-matching characters in your pattern. These special characters are:

- \*           Matches any string, including the null string.
- ?           Matches any one character.
- [ . . . ]   Matches any one of the characters enclosed in square brackets.
- [! . . . ]   Matches any character *other than* one of the characters that follow the exclamation mark within square brackets.

Inside square brackets, a pair of characters separated by a - (minus) specifies a set of all characters lexically within the inclusive range of that pair, according to the current collating sequence (see “**ctab**” on page 257). The **NLCTAB** environment variable controls the collating sequence.

The current collating sequence may group characters into *equivalence classes* for the purpose of defining the end points of a range of characters. For example, if the collating sequence defines the lexical order to be AaBbCc . . . and groups upper- and lower-case characters into equivalence classes, then all the following have the same effect: [a-c], [A-C], [a-C], and [A-c].

---

## Japanese Language Support Information

A collating sequence in Japanese Language Support does not define equivalence classes for use in range expressions. To avoid unpredictable results when using a range expression to match a class of characters, use a *character class expression* rather than a standard range expression. For information about matching file names using character class expressions, see “File-name Substitution in Japanese Language Support” on page 917. See also the discussion of this topic included in the description of the command “File Name Substitution” on page 4.

---

Pattern matching has some restrictions. If the first character of a file name is a . (dot), it can be matched only by a pattern that literally begins with a dot. For example, \* matches the file names `myfile` and `yourfile` but not the file names `.myfile` and `.yourfile`. To match these files, use a pattern such as the following:

```
.*file
```

If a pattern does not match any file names, then the pattern itself is returned as the result of the attempted match.

---

File and directory names should not contain the characters `*`, `?`, `[`, or `]` because they can cause infinite recursion (that is, infinite loops) during pattern-matching attempts.

---

## Japanese Language Support Information

### File-name Substitution in Japanese Language Support

You can also use the following notation to match file names within a range indication:

`[:charclass:]`

This format instructs the system to match any single character belonging to *class*; the defined classes correspond to **ctype** subroutines. Following are the names of these classes:

alnum	jalpha
alpha	jdigit
digit	jhira
lower	jkanji
print	jkata
punct	jparen
space	jpunct
upper	jspace
xdigit	jxdigit

For example, the expression that matches any single kanji character would be the following:

`[[:jkanji:]]`

For additional information about character class expressions, see the discussion of this topic included in the description of the command “**ed**” on page 371.

---

## Shell Variables and Command-Line Substitutions

The shell has several mechanisms for creating variables (assigning a string value to a name). Certain variables, **positional parameters** and **keyword parameters**, are normally set only on a command line. Other variables are simply names to which you or the shell can assign string values.

## *Positional Parameters*

When you run a shell procedure, the shell implicitly creates positional parameters that reference each word on the command line by its position on the command line. The word in position 0 (the procedure name), is called \$0, the next word (the first parameter) is called \$1, and so on up to \$9. To refer to command line parameters numbered higher than 9, use the built-in **shift** command (page 934).

You can also assign values to these positional parameters explicitly by using the built-in **set** command (page 933).

### **Notes:**

1. When an argument for a position is not specified, its positional parameter is set to null.
2. Positional parameters are global and can be passed to nested shell procedures.

## *User-defined Variables*

The shell also recognizes alphanumeric variables to which string values can be assigned. You assign a string value to a **name**, as follows:

```
name=string
```

A name is a sequence of letters, digits, and underscores that begins with an underscore or a letter. To use the value that you have assigned to a variable, add a \$. (dollar sign) to the beginning of its name. Thus \$*name* yields the value *string*. Note that no blanks surround the = (equal sign) in an assignment statement. (Positional parameters cannot appear in an assignment statement; they can only be set as described earlier.) You can put more than one assignment on a command line, but remember: the shell performs the assignments from right to left.

If you surround *string* with quotation marks, either double or single (" " ' '), the shell does not treat blanks, tabs, semicolons, and new-line characters within it as word delimiters but imbeds them literally in the string.

If you surround *string* with double quotation marks (" "), the shell still recognizes variable names in the string and performs **variable substitution**; that is, it replaces references to positional parameters and other variable names that are prefaced by \$ with their corresponding values, if any. The shell also performs **command substitution** (see "Command Substitution" on page 925) within strings that are surrounded by double quotation marks.

If you surround *string* with single quotation marks ( ' '), the shell does no variable or command substitution within the string. The following sequence illustrates this difference:

```
You: stars=*****
      asterisks1="Add $stars"
      asterisks2='Add $stars'
      echo $asterisks1
System: Add *****
You: echo $asterisks2
System: Add $stars
```

The shell does not reinterpret blanks in assignments after variable substitution (see “Blank Interpretation” on page 930). Thus the following assignments result in `$first` and `$second` having the same value:

```
first='a string with embedded blanks'
second=$first
```

When you reference a variable, you can enclose the variable name (or the digit designating a positional parameter) in `{ }` (braces) to delimit the variable name from any following string. In particular, if the character immediately following the name is a letter, digit, or underscore and the variable is not a positional parameter, then the braces are required:

```
You: a='This is a'
      echo "${ a } n example"
Display: This is an example
You: echo "$a test"
Display: This is a test
```

**Note:** The `{ }` operator requires a space following the opening brace and a space preceding the closing brace.

See “Conditional Substitution” on page 920 for a different use of braces in variable substitutions.

## *A Command's Environment*

All the variables (with their associated values) that are known to a command at the beginning of its execution constitute its *environment*. This environment includes variables that a command inherits from its parent process and variables specified as keyword parameters on the command line that calls the command.

The shell passes to its child processes the variables that have been named as arguments to the built-in **export** command. **export** places the named variables in the environments of both the shell and all its future child processes.

Keyword parameters are variable-value pairs that appear in the form of assignments, normally before the procedure name on a command line (but see also the **-k** flag on page 933). Such variables are placed in the environment of the procedure being called.

For example, given the following simple procedure that echoes the values of two variables (saved in a command file named `key_command`):

```
# key_command
echo $a $b
```

the following command lines produce the output shown:

```
You: a=key1 .b=key2 key_command
Display: key1 key2
You: a=tom b=john key_command
Display: tom john
```

A procedure's keyword parameters are not included in the parameter count stored in `$#`.

A procedure can access the values of any variables in its environment; however, if it changes any of these values, these changes are not reflected in the shell environment. They are local to the procedure in question. To place these changes in the environment that the procedure passes to its child processes, you must export these values within that procedure.

To obtain a list of variables that have been made exportable from the current shell, enter:

```
export
```

(You will also get a list of variables that have been made **readonly**.) To get a list of name-value pairs in the current environment, enter:

```
env
```

### ***Conditional Substitution***

Normally, the shell replaces `$variable` with the string value assigned to `variable`, if there is one. However, there is a special notation that allows ***conditional substitution***, depending on whether the variable is set and/or not null. By definition, a variable is ***set*** if it has ever been assigned a value. The value of a variable can be the null string, which you can assign to a variable in any one of the following ways:

```
A=
bcd=""
Efg=''
set '' ""
```

The first three of these examples assign the null string to each of the corresponding variable names. The last example sets the first and second positional parameters to the null string and unsets all other positional parameters.

The following is a list of the available expressions you can use to perform conditional substitution:

- `${ variable-string }` If the variable is set, substitute the value of *variable* in place of this expression. Otherwise, replace this expression with the value of *string*.
- `${ variable:-string }` If the variable is set and is not null, substitute the value of *variable* in place of this expression. Otherwise, replace this expression with the value of *string*.
- `${ variable=string }` If the variable is set, substitute the value of *variable* in place of this expression. Otherwise, set *variable* to *string* and then substitute the value of the *variable* in place of this expression. You cannot assign values to positional parameters in this fashion.
- `${ variable:=string }` If the variable is set and is not null, substitute the value of *variable* in place of this expression. Otherwise, set *variable* to *string* and then substitute the value of the *variable* in place of this expression. You cannot assign values to positional parameters in this fashion.
- `${ variable?string }` If the variable is set, substitute the value of *variable* in place of this expression. Otherwise, display a message of the form:
- variable:* *string*
- and exit from the current shell (unless the shell is the login shell).  
If you do not specify *string*, the shell displays the following message:
- variable:* parameter null or not set
- `${ variable:?string }` If the variable is set and not null, substitute the value of *variable* in place of this expression. Otherwise, display a message of the form:
- variable:* *string*
- and exit from the current shell (unless the shell is the login shell).  
If you do not specify *string*, the shell displays the following message:
- variable:* parameter null or not set
- `${ variable+string }` If the variable is set, substitute the value of *string* in place of this expression. Otherwise, substitute the null string.
- `${ variable:+string }` If the variable is set and not null, substitute the value of *string* in place of this expression. Otherwise, substitute the null string.

In conditional substitution, the shell does not evaluate *string* until it uses it as a substituted string, so that, in the following example, the shell executes the **pwd** command only if **d** is not set or is null:

```
echo ${ d:-`pwd` }
```

### *Variables Used by the Shell*

The shell uses the following variables. The shell sets some of them, and you can set or reset all of them:

- CDPATH** The search path for the **cd** (change directory) command (see the **PATH** variable in the following list for an explanation of search paths).
- HOME** The name of your *login directory*, the directory that becomes the current directory upon completion of a login. The **login** program initializes this variable. The **cd** command uses the value of **\$HOME** as its default value. If you use this variable in your shell procedures rather than using an explicit full path name, your procedures run even if your login directory is changed or if another user runs them.
- LIBPATH** The search path for shared libraries.
- LOGNAME** Your login name, marked **readonly** in the */etc/profile* file.
- MAIL** The path name of the file used by the mail system to detect the arrival of new mail. If **MAIL** is set, the shell periodically checks the modification time of this file and displays the value of **\$MAILMSG** if this time changes and the length of the file is greater than zero.
- You should set **MAIL** in your *.profile* file. The value normally assigned to it by users of the **mail** command is **/usr/mail/\$LOGNAME**.
- MAILCHECK** The number of seconds that the shell lets elapse before checking again for the arrival of mail in the files specified by the **MAILPATH** or **MAIL** parameters. The default value is 600 seconds (10 minutes). If you set **MAILCHECK** to 0, the shell checks before each prompt.
- MAILPATH** A colon-separated list of file names (see **PATH**). If you set this parameter, the shell informs you of the arrival of mail in any of the files specified in the list. You can follow each file name by a **%** (percent sign) and a message to be displayed when mail arrives. Otherwise, the shell uses the value of **MAILMSG** or by default **"you have mail"**.
- Note:** When **MAILPATH** is set, these files are checked instead of the file set by **MAIL**. To check the files set by **MAILPATH** and the file set by **MAIL**, specify the **MAIL** file in your list of **MAILPATH** files.

---

<b>MAILMSG</b>	The mail notification message. If you explicitly set <b>MAILMSG</b> to a null string ( <code>MAIL=""</code> ), no message is displayed.
<b>NLCTAB</b>	Defines the collating sequence to use when sorting names and when character ranges occur in patterns. If absent, it may be taken from the parameter <b>NLFILE</b> . If both are absent, the American English collating sequence is used.

---

### Japanese Language Support Information

When Japanese Language Support is installed, the default collating sequence, based on the character's value, is used. See "Overview of International Character Support" in *Managing the AIX Operating System* for further information.

---

<b>PATH</b>	<p>An ordered list of directory path names separated by colons. The shell searches these directories in the specified order when it looks for commands. A null string anywhere in the list represents the current directory.</p> <p><b>PATH</b> is normally initialized in the <code>/etc/profile</code> file, usually to <code>/bin:/usr/bin:/etc::</code>. You can reset this variable to suit your own needs. Thus if you wish to search your current directory first, rather than last, you would enter:</p> <pre>PATH=:/bin:/usr/bin:/etc</pre> <p>where, by definition, a null string is assumed in front of the leading colon. If you have a personal directory of commands (say, <code>\$HOME/bin</code>) that you want searched before the standard system directories, set your <b>PATH</b> as follows:</p> <pre>PATH=\$HOME/bin:/bin:/usr/bin:/etc::</pre> <p>The best place to set your <b>PATH</b> to something other than the default value is in your <code>.profile</code> file (see "The <code>.profile</code> File" on page 915). You cannot reset <b>PATH</b> if you are executing commands under the restricted shell.</p>
<b>PS1</b>	The string to be used as the primary system prompt. An interactive shell displays this prompt string when it expects input. The default value of <b>PS1</b> is "\$" (a \$ followed by a blank).
<b>PS2</b>	The value of the secondary prompt string. If the shell expects more input when it encounters a new-line character in its input, it prompts with the value of <b>PS2</b> . The default value of <b>PS2</b> is ">" (a > followed by a blank).

<b>IFS</b>	The characters that are <i>internal field separators</i> (the characters that the shell uses during blank interpretation, see “Blank Interpretation” on page 930). The shell initially sets <b>IFS</b> to include the blank, tab, and new-line characters.
<b>SHACCT</b>	The name of a file that you own. If this parameter is set, the shell writes an accounting record in the file for each shell script executed. You can use accounting programs such as <b>acctcom</b> and <b>acctcms</b> to analyze the data collected.
<b>SHELL</b>	A path name whose simple part (the part after the last /) contains the letter r if you want the shell to become restricted when invoked. This should be set and exported by the <b>\$HOME/.profile</b> file of each restricted login.
<b>TIMEOUT</b>	A number of minutes. After the shell displays its prompt, you have <b>TIMEOUT</b> minutes to enter a command. If you fail to do so, the shell exits; in the login shell, such an exit is a logoff. Setting <b>TIMEOUT</b> to 0 inhibits automatic logoff.

### *Predefined Special Variables*

Several variables have special meanings; the following are set *only* by the shell:

- \$#** The number of positional parameters passed to the shell, not counting the name of the shell procedure itself. **\$#** thus yields the number of the highest-numbered positional parameter that is set. One of the primary uses of this variable is to check for the presence of the required number of arguments.
- \$?** The exit value of the last command executed. Its value is a decimal string. Most UNIX commands return 0 to indicate successful completion. The shell itself returns the current value of **\$?** as its exit value.
- \$\$** The process number of the current process. Because process numbers are unique among all existing processes, this string of up to five digits is often used to generate unique names for temporary files. The following example illustrates the recommended practice of creating temporary files in a directory used only for that purpose:

```
temp=$HOME/temp/$$
ls >$temp
.
.
rm $temp
```
- \$!** The process number of the last process run in the background (using the **&** terminator). Again, this is a string of up to five digits.

`$-` A string consisting of the names of the execution flags (page 933) currently set in the shell.

## ***Command Substitution***

To capture the output of any command as an argument to another command, place that command line within grave accents (`` ``). This concept is known as command substitution. The shell first executes the command or commands enclosed within the grave accents, and then replaces the whole expression, grave accents and all, with their output. This feature is often used in assignment statements:

```
today=`date`
```

This statement assigns the string representing the current date to the variable `today`. The following assignment saves, in the variable `files`, the number of files in the current directory:

```
files=`ls | wc -l`
```

You can enclose any command that writes to standard output in grave accents. You can nest command substitutions as long as you quote the inside sets of grave accents with a preceding `\` (backslash):

```
logmsg=`echo Your login directory is `pwd``
```

You can also give values to shell variables indirectly by using the built-in **read** command. **read** takes a line from standard input (usually your keyboard), and assigns consecutive words on that line to any variables named. For example,

```
read first init last
```

will take an input line of the form:

```
J. Q. Public
```

and have the same effect as if you had typed:

```
first=J.   init=Q.   last=Public
```

**read** assigns any excess words to the last variable.

## ***Quoting Mechanisms***

Many characters have a special meaning to the shell; sometimes you want to conceal that meaning. Single (`' '`) and double (`" "`) quotation marks surrounding a string or a backslash (`\`) before a single character provide this function in somewhat different ways.

Within single quotation marks, all characters (except the single quotation character itself), are taken literally, with any special meaning removed. Thus:

```
stuff='echo $? $*; ls * | wc'
```

results only in the literal string `echo $? $*; ls * | wc` being assigned to the variable `stuff`; the `echo`, `ls`, and `wc` commands are not executed, nor are the variables  `$?`  and  `$*`  and the special character  `*`  expanded by the shell.

Within double quotation marks, the special meaning of certain characters (the  `$` ,  `'` , and  `"` ) does persist, while all other characters are taken literally. Thus, within double quotation marks, command and variable substitution takes place. In addition, the quotation marks do not affect the commands within a command substitution that is part of the quoted string, so characters there retain their special meanings.

Consider the following sequence:

```
You: ls *
Display: file1
        file2
        file3
You: message="This directory contains `ls * ` "
     echo $message
Display: This directory contains file1 file2 file3
```

This shows that the  `*`  special character inside the command substitution was expanded.

To hide the special meaning of  `$` ,  `'` , and  `"`  within double quotation marks, precede these characters with a  `\`  (backslash). Outside of double quotation marks, preceding a character with  `\`  is equivalent to placing it within single quotation marks. Hence, a  `\`  immediately preceding the new-line character (that is, a  `\`  at the end of the line) hides the new-line character and allows you to continue the command line on the next physical line.

## Redirection of Input and Output

In general, most commands do not know or care whether their input or output is associated with the keyboard, the display screen, or a file. Thus a command can be used conveniently either at the keyboard or in a pipeline.

## *Standard Input and Standard Output*

When a command begins running, it usually expects that three files are already open: ***standard input***, ***standard output***, and ***diagnostic output*** (sometimes called ***error output*** or ***standard error output***). A number called a ***file descriptor*** is associated with each of these files as follows:

```
File descriptor 0  Standard input
File descriptor 1  Standard output
File descriptor 2  Diagnostic (error) output
```

---

A child process normally inherits these files from its parent; all three files are initially assigned to the work station (0 to the keyboard, 1 and 2 to the display). The shell permits them to be redirected elsewhere before control is passed to a command. Any argument to the shell in the form `<file` or `>file` opens the specified file as the standard input or output, respectively. In the case of output, this process destroys the previous contents of *file*, if it already exists. An argument in the form `>>file` directs the standard output to the end of *file*, thus allowing you to add data to it without destroying its existing contents. If *file* does not exist, the shell creates it.

Such redirection arguments are subject only to variable and command substitution; neither blank interpretation nor pattern matching of file names occurs after these substitutions. Thus:

```
echo 'this is a test' > *.ggg
```

produces a one-line file named `*.ggg` (a disastrous name for a file), and:

```
cat < ?
```

produces an error message, unless you have a file named `?` (also a bad choice for a file name).

### ***Diagnostic and Other Output***

Diagnostic output from UNIX commands is normally directed to the file associated with file descriptor 2. You can redirect this error output to a file by immediately preceding either output redirection arrow (`>` `>>`) with a 2 (the number of the file descriptor). For example, the following line adds error messages from the `cc` command to the file `ERRORS`:

```
cc testfile.c 2>> ERRORS
```

Note that there must be no blanks between the file descriptor and the redirection symbol; otherwise, the shell interprets the number as a separate argument to the command.

You can also use this method to redirect the output associated with any of the first 10 file descriptors (numbered 0 through 9) so that, for instance, if a command (`cmd`) writes to file descriptor 9 (although this is not a recommended programming habit), you can capture that output in a file `savedata` as follows:

```
cmd 9> savedata
```

If a command writes to more than one output, you can independently redirect each one. Suppose that a command directs its standard output to file descriptor 1, directs its error output to file descriptor 2, and builds a data file on file descriptor 9. The following command line redirects each of these outputs to a different file:

```
cmd > standard 2> error 9> data
```

### ***Inline Input Documents***

Upon seeing a command line of the form:

```
cmd << eofstring
```

where *eofstring* is any string that does not contain any pattern-matching characters, the shell takes the subsequent lines as the standard input of `cmd` until it reads a line consisting of only *eofstring* (possibly preceded by one or more tab characters). The lines between the first *eofstring* and the second are frequently referred to as a ***here document***. If a - (minus) immediately follows the <<, the shell strips leading tab characters from each line of the input document before it passes the line to the command.

The shell creates a temporary file containing the input document and performs variable and command substitution on its contents before passing it to the command. It performs pattern matching on file names that are a part of command lines in command substitutions. If you want to prohibit all substitutions, quote any character of *eofstring*:

```
cmd << \eofstring
```

The here document is especially useful for a small amount of input data that is more conveniently placed in the shell procedure rather than kept in a separate file (such as editor "scripts"). For instance, you could enter:

```
cat <<- xyz
```

```
This message will be shown on the  
display with leading tabs removed.
```

```
xyz
```

This feature is most useful in shell procedures. Note that inline input documents *cannot* appear within grave accents (command substitution).

### ***Input and Output Redirection Using File Descriptors***

As discussed previously, a command occasionally directs output to some file associated with a file descriptor other than 1 or 2. The shell also provides a mechanism for creating an output file associated with a particular file descriptor. By entering:

```
fd1>&fd2
```

where *fd1* and *fd2* are valid file descriptors, you can direct the output that would normally be associated with file descriptor *fd1* to the file associated with *fd2*. The default value for *fd1* and *fd2* is 1 (standard output). If, at execution time, no file is associated with *fd2*, then the redirection is void. The most common use of this mechanism is to ***direct*** standard error output to the same file as standard output, as follows:

```
cmd 2>&1
```

If you want to **redirect** both standard output and standard error output to the same file, enter:

```
cmd > file 2>&1
```

The order here is significant. First, the shell associates file descriptor 1 with `file`; then it associates file descriptor 2 with the file that is currently associated with file descriptor 1. If you reverse the order of the redirections, standard error output will go to the display and standard output would go to `file` because at the time of the error output redirection, file descriptor 1 was still associated with the display.

You can also use this mechanism to redirect standard input. You could enter:

```
fda<&fdb
```

to cause both file descriptors to be associated with the same input file. For commands that run sequentially, the default value of `fda` and `fdb` is 0 (standard input). For commands that run asynchronously (commands terminated by `&`), the default value of `fda` and `fdb` is `/dev/null`. Such input redirection is useful for commands that use two or more input sources.

### ***Summary of Redirection Options***

The following can appear anywhere in a simple command or can precede or follow a command, but they are not passed to the command:

- |                                 |   |
|---------------------------------|---|
| <code>&lt; file</code>          | Use <i>file</i> as standard input.  |
| <code>&gt; file</code>          | Use <i>file</i> as standard output. Create the file if it does not exist; otherwise truncate it to zero length.   |
| <code>&gt;&gt; file</code>      | Use <i>file</i> as standard output. Create the file if it does not exist; otherwise add the output to the end of the file.  |
| <code>&lt;&lt; [-]eofstr</code> | Read as standard input all lines from <i>eofstr</i> up to a line containing only <i>eofstr</i> or up to an end-of-file character. If any character in <i>eofstr</i> is quoted, the shell does not expand or interpret any characters in the input lines; otherwise, it performs variable and command substitution and ignores a quoted new-line character ( <code>\new-line</code> ). Use a <code>\</code> to quote characters within <i>eofstr</i> or within the input lines.<br><br>If you add a - (minus) to <code>&lt;&lt;</code> then all leading tabs are stripped from <i>eofstr</i> and from the input lines. |
| <code>&lt; &amp;digit</code>    | Associate standard input with file descriptor <i>digit</i> .  |
| <code>&gt; &amp;digit</code>    | Associate standard output with file descriptor <i>digit</i> .   |
| <code>&lt; &amp;-</code>        | Close standard input.   |

>&- Close standard output.

The restricted shell does not allow the redirection of output.

## Blank Interpretation

After the shell performs variable and command substitution, it scans the results for internal field separators (those defined in the shell variable **IFS**, see page 924). It splits the line into distinct words at each place it finds one of these characters. It retains explicit null arguments (" " ' ') and discards implicit null arguments (those resulting from parameters that have no values).

## Control Commands

The shell provides several flow control commands that are useful in creating shell procedures:

**for** *name* [ **in** *word* . . . ]

**do** *list*

**done**

For each *word*, sets *name* to *word* and executes the commands in *list*. If you omit **in** *word* . . . , then the **for** command executes *list* for each positional parameter that is set. Execution ends when there are no more words in the word list.

**case** *word* **in**

*pattern* [**!***pattern*] . . . ) *list*;;

[ .

·  
·

*pattern* [**!***pattern*] . . . ) *list*;;]

**esac**

Executes the commands in the *list* associated with the first *pattern* that matches *word*. Uses the same character-matching notation in patterns that you use for file-name substitution (see “File-name Substitution” on page 916), except that you do not need to match explicitly a slash, a leading dot, or a dot immediately following a slash.

**if** *list*

**then** *list*

[**elif** *list*] . . .

[**else** *list*]

**fi**

Executes the *list* following the **if** keyword. If it returns a zero exit value, execute the *list* following the first **then**. Otherwise, execute the *list* following **elif** (if there is an **elif**), and if its exit value is zero, execute the next **then**. Failing that, execute the *list* following the **else**.

If no **else** *list* or **then** *list* is executed, the **if** command returns a zero exit value.

**while** *list*  
**do** *list*  
**done**

Executes the *list* following the **while**. If the exit value of the last command in the list is zero, executes the *list* following **do**. Continue looping through the *lists* until the exit value of the last command in the **while** *list* is nonzero. If no commands in the **do** *list* are executed, the **while** command returns a zero exit value.

**until** *list*  
**do** *list*  
**done**

Executes the *list* following the **until**. If the exit value of the last command in the list is nonzero, executes the *list* following **do**. Continues looping through the *lists* until the exit value of the last command in the **until** *list* is zero. If no commands in the **do** *list* are executed, the **until** command returns a zero exit value.

(*list*)

Executes the commands in *list* in a subshell.

{ *list*; }

Executes the commands in *list* in the current shell process (does not spawn a subshell).

*name* () { *list*; }

Defines a function that is referenced by *name*. The body of the function is the *list* of commands between the braces.

The following words are recognized only as the first word of a command and when not quoted:

**if then else elif fi case esac for while until do done { }**

## Built-in Commands

- :** Does nothing. This null command returns a zero exit value.
- . *file*** Reads and executes commands from *file* and returns. Does not spawn a subshell. The search path specified by **PATH** is used to find the directory containing *file*.
- break [*n*]** Exits from the enclosing **for**, **while**, or **until** loop, if any. If *n* is specified, then breaks *n* levels.
- continue [*n*]** Resumes the next iteration of the enclosing **for**, **while**, or **until** loop. If *n* is specified, resumes at the *n*th enclosing loop.
- cd [*dir*]** Changes the current directory to *dir*. The value of the shell variable **HOME** is the default *dir*. The shell variable **CDPATH** defines the search path for the directory containing *dir*. Alternative directory names appear in a colon-separated list. A null path name specifies the current directory (which is the default path). This null path name can

appear immediately after the equal sign in the assignment or between the colon delimiters anywhere else in the path list. If *dir* begins with a slash, the shell does not use the search path. Otherwise, the shell searches each directory in the path. **cd** cannot be executed by the restricted shell.

- dirstyle** Indicates whether directories should be interpreted in either raw or System V format. A + flag indicates path names from remote file systems should be converted to System V format. A - flag indicates path names from remote file systems should not be converted to System V.
- Note:** Use this only when reading the contents of a directory that is not formatted in System V. Otherwise, use **opendir**, **readdir**, and **closedir** library functions.
- echo** [*arg* . . . ] Writes arguments to standard output. See “**echo**” on page 369 for a discussion of its usage and parameters.
- eval** [*arg* . . . ] Reads arguments as input to the shell and executes the resulting command(s).
- exec** [*arg* . . . ] Executes the command specified by argument in place of this shell without creating a new process. Input and output arguments can appear and, if no other arguments appear, cause the shell input or output to be modified (not a good idea with your login shell).
- exit** [*n*] Causes a shell to exit with the exit value specified by *n*. If you omit *n*, the exit value is that of the last command executed (**Ctrl-D** will also cause a shell to exit). The value of *n* can be from 0 to 255, inclusive.
- export** [*name* . . . ] Marks the specified *names* for automatic export to the environments of subsequently executed commands. If you do not specify *names*, **export** displays a list of all names that are exported in this shell. You *cannot* export function names.
- hash** [-r] [*name* . . . ] For each *name*, finds and remembers the location in the search path of the command specified by *name*. The -r flag causes the shell to forget all locations. If you do not specify the flag or any *names*, the shell displays information about the remembered commands. In this information, *hits* is the number of times a command has been run by the shell process. *Cost* is a measure of the work required to locate a command in the search path. There are certain situations that require that the stored location of a command be recalculated (for example, the location of a relative path name when the current directory changes). Commands for which that might be done are indicated by an asterisk next to the hits information. Cost is incremented when the recalculation is done.

- 
- newgrp** [*arg* . . . ]  
Executes the **newgrp** command in the current shell process. See “**newgrp**” on page 689 for a discussion of command options.
- pwd**  
Displays the current directory. See “**pwd**” on page 800 for a discussion of command options.
- read** [*name* . . . ]  
Reads one line from standard input. Assigns the first word in the line to the first *name*, the second word to the second *name*, and so on, with leftover words assigned to the last *name*. This command returns a 0 unless it encounters an end of file.
- readonly** [*name* . . . ]  
Marks the specified *names* **readonly**. The values of these *names* cannot be reset. If you do not specify any *names*, **readonly** displays a list of all **readonly** names.
- return** [*n*]  
Cause a function to exit with a return value of *n*. If you do not specify *n*, the function returns the status of the last command executed in that function. This command is valid only when executed within a shell function.
- set** [*flag* . . . [*arg*] . . . ]
- a Marks for export all variables that are modified or changed.
  - e Exits immediately if a command exits with a nonzero exit value.
  - f Disables file-name substitution.
  - h Locates and remembers the commands called within functions as the functions are defined (normally these commands are located when the function is executed—see the **hash** command on page 932).
  - k Places all keyword parameters in the environment for a command, not just those that precede the command name.
  - n Reads commands but do not execute them.
  - t Exits after reading and executing one command.
  - u Treats an unset variable as an error when performing variable substitution.
  - v Displays shell input lines as they are read.
  - x Displays commands and their arguments as they are executed.
  - Does not change any of the flags. This is useful in setting \$1 to a string beginning with a - (minus).

Using a + (plus) rather than a - (minus) unsets flags. You can also specify these flags on the shell command line. The special variable `$-` contains the current set of flags.

Any arguments to **set** are positional parameters and are assigned, in order, to `$1`, `$2`, and so on. If you do not specify flags or parameters, **set** displays all names.

- shift** [*n*] Shifts command line arguments to the left; that is, reassign the value of the positional parameters by discarding the current of value of `$1` and assigning the value of `$2` to `$1`, of `$3` to `$2`, and so on. If there are more than 9 command line arguments, the tenth is assigned to `$9` and any that remain are still unassigned (until after another **shift**). If there are 9 or fewer arguments, a **shift** unsets the highest-numbered positional parameter.
- `$0` is never shifted. The command **shift** *n* is a shorthand notation for *n* consecutive shifts. The default value of *n* is 1.
- test** *expr* | [*expr*] Evaluates conditional expressions. See “**test**” on page 1064 for a discussion of command options.
- times** Displays the accumulated user and system times for processes run from the shell.
- trap** [*arg*] [*n*] . . . Runs the command specified by *arg* when the shell receives signal(s) *n*. (Note that the shell scans *parm* once when the trap is set and once when the trap is taken). **trap** commands are executed in order of signal number. Any attempt to set a trap on a signal that was ignored on entry to the current shell is ineffective.
- If you do not specify an *arg*, then all trap(s) *n* are reset to their current values. If *arg* is the null string, then this signal is ignored by the shell and by the commands it invokes. If *n* is 0, then *arg* is executed on exit from the shell. If you specify no *arg* and no *n*, **trap** displays a list of commands associated with each signal number.
- type** [*name* . . . ] For each *name*, indicates how the shell would interpret it as a command name.
- ulimit** [-**b** [*m*]] [-**f**] [*n*] [-**s** [*m*]] Sets or queries size limits. The **-b** flag sets the break value to *m*. This limits the size of data segment to *m* pages. If you specify **-b** with no *m*, **ulimit** displays the current break value. The **-f** flag imposes a size limit of *n* blocks on files written by the child processes (files of any size can be read). Without *n*, **ulimit** displays the current limit (this is the default action of **ulimit**).

**Notes:**

1. Since the shell rounds *n* down to the nearest cluster size, it is best to make it a multiple of 4 (for example, specifying values of 1, 2, or 3 for *n* all result in a size limit of 0).
2. Any user can decreased this limit, but only a user operating with superuser authority can increase the limit.

The **-s** flag imposes a size limit of *m* pages on the stack. If you specify **-s** with no *m*, **ulimit** displays the current stack size limit.

<b>umask</b> [ <i>nnn</i> ]	Sets the user file-creation mask to <i>nnn</i> (see the <b>umask</b> system call). If you omit <i>nnn</i> , <b>umask</b> displays the current value of the mask.
<b>unset</b> [ <i>name</i> . . . ]	For each <i>name</i> , removes the corresponding variable or function. The variables <b>PATH</b> , <b>PS1</b> , <b>PS2</b> , <b>MAILCHECK</b> and <b>IFS</b> cannot be unset.
<b>wait</b> [ <i>n</i> ]	Waits for the child process whose process number is <i>n</i> to end and reports its termination status. If you do not specify <i>n</i> , then the shell waits for all currently active child processes and the return value is 0.

**Running the Shell**

The **sh** command can be run either as a *login shell* or as a login shell subshell under the login shell. Only the **login** command can call **sh** as a login shell. It does this by using a special form of the **sh** command name: **-sh**. When called with an initial - (minus), the shell first reads and runs commands found in the system **profile** file and your **\$HOME/.profile**, if one exists. It then accepts commands as described in the following discussion of flags. Once logged in and working under a login shell, you can call **sh** with the command name **sh**. This command runs a subshell, a second shell running as a child of the login shell.

**Restricted Shell**

The restricted shell, **Rsh**, is used to set up login names and environments whose capabilities are more controlled than those of the standard shell. The actions of **Rsh** are identical to those of **sh**, except that the following are not allowed:

- Changing directory (see “**cd**” on page 150)
- Setting the value of **\$PATH**
- Specifying path or command names containing /
- Redirecting output (**>** and **>>**).

The restrictions above are enforced after **.profile** is interpreted, meaning that the restrictions can override settings originally set in **.profile**.

## sh

---

When a command to be run is found to be a shell procedure, **Rsh** starts **sh** to run it. Thus, it is possible to provide to the end-user shell procedures that have access to the full power of the standard shell, while imposing a limited menu of commands; this scheme assumes that the end-user does not have write and run permissions in the same directory.

The net effect of these rules is that the writer of the **.profile** has complete control over user actions, by performing setup actions and leaving the user in an appropriate directory (probably not the login directory).

When called with the name **-Rsh**, **Rsh** reads the user's **.profile** (from `$HOME/.profile`). It acts as the standard **sh** while doing this, except that an interrupt causes an immediate exit instead of a return to command level.

## Flags

The following flags are interpreted by the shell only when you call it. Note that unless you specify either the **-c** or **-s** flag, the shell assumes that the next parameter is a command file (shell procedure). It passes anything else on the command line to that command file (see "Positional Parameters" on page 918).

**-c** *cmdstring*

Runs commands read from *cmdstring*. The shell does not read additional commands from standard input when you specify this flag.

**-i** Makes the shell interactive, even if input and output are not from a work station. In this case the shell ignores the **TERMINATE** signal (so that **kill 0** does not stop an interactive shell) and traps an **INTERRUPT** (so that you can interrupt **wait**). In all cases, the shell ignores the **QUIT** signal. (See the **signal** system call in *AIX Operating System Technical Reference* and "**kill**" on page 552 for more information about signals.)

**-r** Creates a restricted shell (the same as running **Rsh**).

**-s** Reads commands from standard input. Any remaining parameters specified are passed as positional parameters to the new shell. Shell output is written to standard error, except for the output of built-in commands (see "Built-in Commands" on page 931).

The remaining flags and parameters are described in the built-in **set** command on page 933.

## Files

`/etc/environment`  
`/etc/profile`  
`$HOME/.profile`  
`/tmp/sh*`  
`/dev/null`

---

## Related Information

The following commands: “**cd**” on page 150, “**echo**” on page 369, “**env**” on page 393, “**login**” on page 584, “**newgrp**” on page 689, “**pwd**” on page 800, “**test**” on page 1064 and “**umask**” on page 1110.

The **dup**, **exec**, **fork**, **fullstat**, **pipe**, **signal**, **ulimit**, and **umask** system calls and the **a.out**, **.profile**, and **environ** files in *AIX Operating System Technical Reference*.

“Using the Shell with Processes” and “Advanced Shell Features” in *Using the AIX Operating System*.

“Overview of International Character Support” in *Managing the AIX Operating System*.

The discussion of Japanese Language Support in *Japanese Language Support User's Guide*.

# shell

---

## shell

---

### Purpose

Executes a shell in a user's login environment

### Syntax



AJ2FL133

### Description

The **shell** command returns a user to an environment that is equivalent to the user's login environment. This command issues a **frevoke** system call to end all processes accessing the port. It re-establishes the terminal modes and environment variables then runs the user's login shell or the specified shell.

Only a person with superuser authority can specify *port* and *shell*. This command ignores any other arguments. When *port* and *shell* are not specified, the shell is found in **/etc/passwd** and the terminal is the one from which this command is issued.

---

#### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

### File

**/etc/passwd** Specifies the user's login shell.

### Related Information

See the section on trusted path management in *Managing the AIX Operating System*.

---

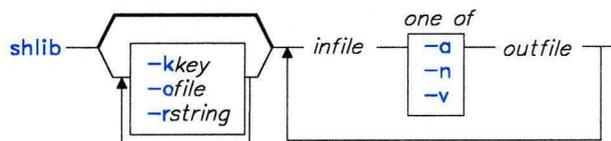
# shlib

---

## Purpose

Creates a shared library.

## Syntax



OL805448

## Description

The **shlib** command creates a shared library from a set of unshared object and/or archive files. The shared library it creates has two parts:

- A single shared library text image that contains the code, and only the code, from all of the input files.
- Modified archive or object files that refer to the text image, each of which corresponds to one of the **shlib** input files.

By default, **shlib** uses the name of the first input file to generate the shared library key. It does this by removing any directory path from the file name and, if the file name does not contain a suffix, adding the suffix **.yyddd**, where **yy** is the last two digits of the current year and **ddd** is the number of the day. **shlib** puts this key in each of the modified output files. By default, it also uses this key as the name of the shared library text image, which contains the shared library key in a form that can be found by the **what** command.

The **shlib** command transforms each input file specified on the command line and copies it to (or verifies it against) an output file. Each output object module differs from the corresponding input object module in that the text portion has been removed and added to the end of the shared library text image. **shlib** also appends the shared library key to each output module and marks its **a.out** header so the **ld** command recognizes that the file refers to a shared library and relocates references appropriately.

Once you create an archive for a shared library, you can use the **ar** command to replace individual object files. The new object files will not refer to the shared library, thus allowing you to patch shared libraries.

## shlib

---

The **shlib** command can process all **cc** and **f77** programs. Other programs must have **KCALL** relocation entries as the only external references within the text portion (see the **a.out** file in *AIX Operating System Technical Reference*). **KCALL** relocation entries are replaced by **balax** instructions to location 0xC00, which contain a code fragment to continue calls across segments. That code requires that register 0 point to the constant pool of the called routine, the first entry of which is the entry point of the invoked routine.

### Flags

- |                          |  |
|--------------------------|--|
| <b>-a</b> <i>outfile</i> | Adds new object modules in archives to <i>outfile</i> . This lets you add functions to a shared library, replacing the shared library text image without relinking programs that refer to it. The entire shared library image must be rebuilt.   |
| <b>-k</b> <i>key</i>     | Uses the shared library <i>key</i> in the output object modules to refer to the shared library text image. If <i>key</i> does not contain a . (period), a suffix in the form <b>.yyddd</b> is added.   |
| <b>-n</b> <i>outfile</i> | Makes a new output file for each input file.   |
| <b>-o</b> <i>file</i>    | Assigns the name <i>file</i> to the shared library text image. <b>shlib</b> always makes a new shared library text image, replacing the old one.   |
| <b>-r</b> <i>string</i>  | Adds <i>string</i> to the end of the shared library key in the <b>what</b> string. This only applies to the text image file.   |
| <b>-v</b> <i>outfile</i> | Verifies components of a changed shared library, thus ensuring that the resulting object files are the same as the files previously created by <b>shlib</b> . Changes can be made to C source code as long as references to external names are not added, deleted, or rearranged and no floating point, string, or static initial values are modified. |

### Examples

1. To create a new shared text image:

```
shlib -kclibs -r"C shared library text" \  
/lib/libc.a -n /lib/libcs.a /lib/librts.a -n /lib/librtss.a
```

This creates shared libraries **libcs.a** and **librtss.a** and the text image file **clibs.86133** (86133 indicates that this file was produced on 13 May 1986).

2. To modify an existing shared text image:

```
shlib -klibs -r"C shared library text" \  
/lib/libc.a -a /lib/libcs.a /lib/libm.a -n /lib/libms.a
```

This updates the shared library `libcs.a`, creates the shared library `libms.a`, and updates the shared text image `clibs.86133`. (`libc.a` may contain new members that are added to the shared library and the text image.)

3. To create a shared text image with a different name than its key:

```
shlib -klibs -octext_image -r"C shared library text" \  
/lib/libc.a -n /lib/libcs.a /lib/librts.a -n /lib/librtss.a
```

This produces a text image file named `ctext_image`. Note that you must use the `-k` flag when you compile programs that use this text image, since the key and the file name are not the same:

```
cc myproj.c -klibs:ctext_image -lrtss -lcs
```

## Related Information

The following command: “**ld**” on page 557.

The **profil** system call, **monitor** subroutine and **a.out** file in *AIX Operating System Technical Reference*.

The discussion of shared libraries in *AIX Operating System Programming Tools and Interfaces*.

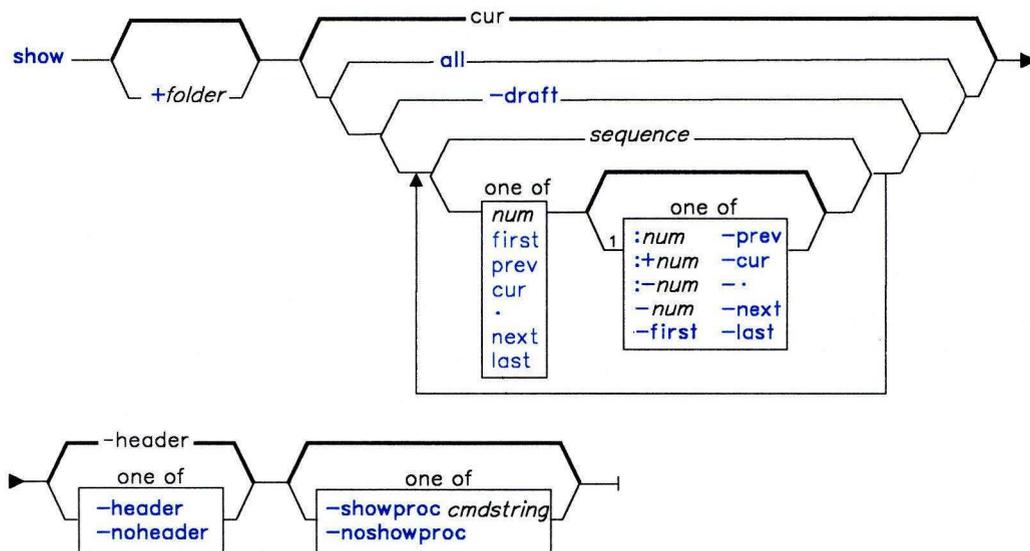
# show

## show

### Purpose

Shows messages.

### Syntax



AJ2FL207

`show` — `-help` — |

AJ2FL208

<sup>1</sup> Do not put a blank between these items.

OL805308

### Description

The **show** command is used to display a list of messages. **show** is part of the MH (Message Handling) package and can be used with other MH and AIX commands.

The **show** command sends a listing of each of the specified messages to standard output. If standard output is not a display, **show** lists each message with a one-line header and two separation lines.

The **show** command invokes a program to perform the listing. The system default is **/bin/pg**. You can define your own default with the **showproc:** entry in **\$HOME/.mh-profile**. If you set **showproc:** entry to **mhl**, **show** calls an internal **mhl** routine instead of the **mhl** command. You can also specify the program to perform a listing in the *cmdstring* of the **-showproc** flag.

The **show** command passes any flags that it does not recognize to the program performing the listing. Thus, you can specify flags for the listing program, as well as the flags described in this command section.

If the **Unseen-Sequence** entry is present in **\$HOME/.mh-profile** and is not empty, **show** removes each of the messages shown from each sequence named by the profile entry.

## Flags

- draft** Shows the file *user\_mh\_directory/draft* if it exists.
- +folder msgs** Specifies the messages that you want to show. *msgs* can be several messages, a range of messages, or a single message. You can use the following message references when specifying *msgs*:
- |             |              |                 |
|-------------|--------------|-----------------|
| <i>num</i>  | <b>first</b> | <b>prev</b>     |
| <b>cur</b>  | .            | <b>next</b>     |
| <b>last</b> | <b>all</b>   | <i>sequence</i> |
- The default message is the current message in the current folder. If several messages are specified, the last message shown becomes the current message.
- header** Displays a one-line description of the message being shown. The description includes the folder name and the message number. If you show more than one message, this flag does not produce message headers. This flag is the default.
- help** Displays help information for the command.
- noheader** Does not display a one-line description of each message being shown.
- noshowproc** Uses **/bin/cat** to perform the listing.
- showproc cmdstring** Uses the specified command string to perform the listing.

## show

---

### Profile Entries

**Current-Folder:** Sets your default current folder.  
**Path:** Specifies your *user\_mh\_directory*.  
**showproc:** Specifies the program used to show messages.  
**Unseen-Sequence:** Specifies the sequences used to keep track of your unseen messages.

### Files

`$HOME/.mh_profile` The MH user profile.  
`user_mh_directory/draft` The draft file.

### Related Information

The following commands: “**mhl**” on page 643, “**next**” on page 694, “**pick**” on page 748, “**prev**” on page 765, “**scan**” on page 871, “**sendmail**” on page 897.

The **mh-format**, **mh-mail**, and **mh-profile** files in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

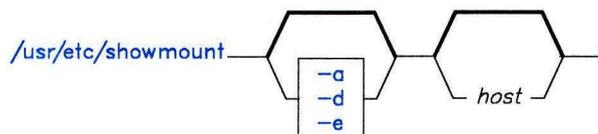
# showmount

---

## Purpose

Shows all file systems that are mounted remotely in an NFS environment.

## Syntax



OL805502

## Description

The **showmount** command lists the clients that have mounted a file system from an NFS server. This information is maintained by the network daemon **mountd** which runs continuously on the server. The daemon saves this information to the file **/etc/rmtab** for use if the server crashes.

The default value for *host* is the value returned by the command **hostname**.

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

## Flags

- a Displays all remote mounts in the format *hostname:directory*, where *hostname* specifies the name of the client and *directory* specifies the root of the mounted file system.
- d Lists directories that have been remotely mounted by clients.
- e Prints the list of exported file systems.

# shutdown

---

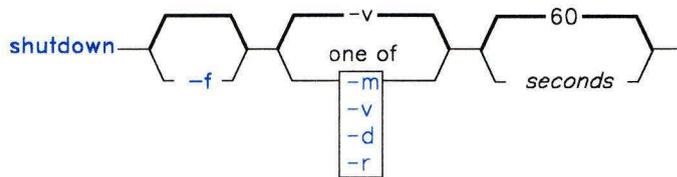
## shutdown

---

### Purpose

Ends system operation.

### Syntax



OL805215

### Description

The **shutdown** command brings the AIX Operating System from distributed mode to multiuser mode, from multiuser mode to maintenance mode, or halts the operating system completely. You can run **shutdown** only if you are a member of the system group or if you have superuser authority.

During the default shutdown, users are notified (by a **wall** command) of the impending system shutdown with the message: THE SYSTEM IS COMING DOWN NOW. However, the shutdown is not complete until the user receives the message: . . . shutdown completed. . . . Do not attempt to reboot the system (**Ctrl-Alt-Pause**) or turn off the system before this message is displayed; otherwise, file system damage may result.

During the shutdown, the **hold** command prevents any new logins. After the specified number of *seconds* (60 by default), the system stops the accounting and error-logging processes. **shutdown** then runs the **killall** command to end any remaining processes and runs the **sync** command to flush all memory resident disk blocks. Finally, it unmounts the file systems and sends the appropriate signal to **init**. These signals are:

<b>SIGINT</b>	Maintenance mode
<b>SIGTERM</b>	Virtual machine halt.

**Note:** Users who have files open on the node that is running **shutdown**, but who are not logged in to that node are not notified about the shutdown.

If there are no other users on the system and you want to shutdown the system quickly, you may want to use **shutdown -f**. This option bypasses messages to users and brings the system down as quickly as possible.

If you request a complete halt to the operating system, **shutdown** kills all processes, unmounts all file systems, and sends **init** the **SIGTERM** signal.

**Warning:** If you are bringing the system down to maintenance mode, you must run **shutdown** from the root directory to ensure that it can cleanly unmount the file systems.

## Flags

- d** Brings the system down from a distributed mode to a multiuser mode.
- f** Does a fast shutdown, bypassing the messages to other users and bringing the system down as quickly as possible. If you do not specify this flag, **shutdown** sends a message to each logged-in user and waits a certain amount of time before bringing the system down, to allow each user to log off cleanly.
- m** Brings the system down to maintenance mode. From maintenance mode, you can return to multiuser mode.
- r** Causes the system to automatically reboot (**Ctrl-Alt-Pause**) after the user receives the message `..shutdown completed`.
- v** Halts the operating system completely.

## Examples

1. To tell the operating system you are about to turn off the machine:

```
shutdown
```

This shuts down the system, waiting 60 seconds before stopping the user processes and the **init** process.

2. To give users and the system more time to finish what they are doing:

```
shutdown -m 120 *
```

This brings the system down from multiuser mode to maintenance mode after waiting 120 seconds.

## Files

```
/etc/shutdown.sh  
/etc/fshutdown.sh
```

## shutdown

---

### Related Information

The following commands: “**acct/\***” on page 13, “**errstop**” on page 404, “**init**” on page 521, “**kill**” on page 552, and “**killall**” on page 555,

The **dsstate** and **signal** system calls in *AIX Operating System Technical Reference*.

---

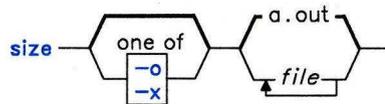
# size

---

## Purpose

Displays the section sizes of common object files.

## Syntax



OL805216

## Description

The **size** command writes to the standard output the number of bytes required by the text, initialized data, and uninitialized data, along with their sum for each *file*. The default *file* is **a.out**.

## Flags

The output is in decimal notation unless you change the output with the following flags:

- o Writes in octal notation.
- x Writes in hexadecimal notation.

## Examples

1. To display the size of **a.out** in decimal:

```
size
```

This displays the size in bytes of the executable file **a.out**. The size of each section of the object file is given, followed by the total. The three sections are program text, data, and bss (uninitialized data). The values are in decimal.

2. To display the size of an object file in octal:

```
size -o driver.o
```

This displays the size of the object file `driver.o` in octal.

## size

---

3. To display the size of several object files in hexadecimal:

```
size -x *.o
```

This displays in hexadecimal the size of each file in the current directory ending with .o.

## Related Information

The following commands: “**ar**” on page 55, “**as**” on page 61, “**cc**” on page 140, “**dump**” on page 366, “**ld**” on page 557, “**nm**” on page 705, and “**strip**” on page 1017.

The discussion of the **a.out** and **ar** files in *AIX Operating System Technical Reference*.

# skulker

---

## Purpose

Cleans up file systems by removing unwanted files.

## Syntax

`skulker`  $\rightarrow$

OL805217

## Description

**Warning:** Because this command file runs with superuser authority and its whole purpose is to remove files, it has the potential for unexpected results. Before installing a new **skulker**, test any additions to its file removal criteria by running it manually using the **xargs -p** command. After you have verified that the new **skulker** removes only the files you want removed, you can install it.

The **skulker** command is a shell command file for periodically purging obsolete or unneeded files from file systems. Candidate files include those in **/tmp**, **.bak** files older than a specified age, and files named **a.out**, **core**, or **ed.hup**.

The **skulker** command is normally invoked daily, often as part of an accounting procedure run by **cron** during off-peak periods. Individual sites should modify **skulker** to suit local needs following the patterns shown in the distributed version. Local users should be made aware of the criteria for automatic file removal.

The **find** and **xargs** commands form a powerful combination for use in **skulker**. Most file selection criteria can be expressed conveniently with **find** expressions. The resulting file list, generated with the **-print** flag of the **find** command, can then be segmented and inserted into **rm** commands using **xargs** to reduce the overhead that would result if each file were deleted with a separate command.

## Related Information

The following commands: “**cron**” on page 220, “**find**” on page 422, “**rm**” on page 833, and “**xargs**” on page 1232.

The section on using the **skulker** in *Managing the AIX Operating System*.

# sleep

---

## sleep

---

### Purpose

Suspends execution for an interval.

### Syntax

```
sleep — seconds —|
```

OL805218

### Description

The **sleep** command suspends execution of a process for the interval specified by *seconds*. *seconds* can range from 1 to 65,536 seconds.

### Examples

1. To run a command after a certain amount of time has passed:

```
echo "SYSTEM SHUTDOWN IN 10 MINUTES!" | wall  
(sleep 300; echo "SYSTEM SHUTDOWN IN 5 MINUTES!" | wall) &  
(sleep 540; echo "SYSTEM SHUTDOWN IN 1 MINUTE!" | wall) &  
(sleep 600; shutdown) &
```

This command sequence warns all users 10 minutes, 5 minutes, and 1 minute before the system is shut down.

2. To run a command at regular intervals:

```
while true  
do  
    date  
    sleep 60  
done
```

This shell procedure displays the date and time once a minute. To stop it, press **INTERRUPT (Alt-Pause)**.

## Related Information

The following commands: “**shutdown**” on page 946 and “**wall**” on page 1208.

The **alarm** system call and the **sleep**, **pause**, and **signal** subroutines in *AIX Operating System Technical Reference*.

# slocal

---

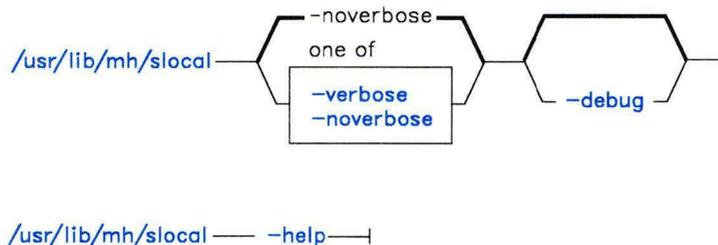
## slocal

---

### Purpose

Processes incoming mail.

### Syntax



AJ2FL259

### Description

The **slocal** command is used to perform a set of actions each time a message is sent to the user. **slocal** is not designed to be run directly by the user; it is designed to be called by the **sendmail** command. The **slocal** command is part of the MH (Message Handling) package and can be used with other MH and AIX commands.

The **sendmail** command invokes the **slocal** command when it encounters the following line in **\$HOME/.forward**:

```
| /usr/lib/mh/slocal
```

For each new incoming message, **slocal** performs the actions specified in the **.maildelivery** file. If **slocal** cannot find the file **\$HOME/.maildelivery**, **slocal** uses the default file **/usr/lib/mh/maildelivery**. If the delivery request fails, **slocal** delivers the message to **/usr/mail/\$USER**.

The actions that can be specified in **.maildelivery** are described in *AIX Operating System Technical Reference* under **mhooks**.

### Flags

<b>-debug</b>	Provides information for debugging.
<b>-help</b>	Displays help information for the command.

- noverbose** Does not display information as the system executes commands in the maldelivery file. This flag is the default.
- verbose** Displays information as the system executes commands in the maldelivery file.

## Files

- |                         |  |
|-------------------------|--|
| /usr/lib/mh/mtstailor   | The MH tailor file.                              |
| /usr/lib/mh/maldelivery | The default MH local delivery instructions file. |
| \$HOME/.maldelivery     | The user's local delivery instructions file.     |
| \$HOME/.forward         | The user's default message filter file.          |

## Related Information

The following commands: “**rcvdist**” on page 808, “**rcvpack**” on page 810, “**rcvstore**” on page 812, “**rcv tty**” on page 815, “**sendmail**” on page 897.

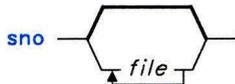
The **mh-format**, **mhook**, **mh-mail**, and **mh-profile** file formats in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

## Purpose

Provides a SNOBOL interpreter.

## Syntax



OL805219

## Description

The **sno** command provides a SNOBOL compiler and interpreter, with some differences from standard SNOBOL. It reads the named *files* and the standard input. It compiles all input through a statement containing the label **end**. The rest is available to **syspit**. The **sno** command differs from SNOBOL in the following ways:

- There are no unanchored searches. To get the same effect:  
a \*\* b                    Unanchored search for *b*  
a \*x\* b = x c            Unanchored assignment.
- There is no back referencing.  
x = "abc"  
a \*x\* x                    Unanchored search for **abc**.
- Function declaration is done at compile time by the use of the (nonunique) label **define**. Execution of a function call begins at the statement following the **define**. Functions cannot be defined at run time, and the use of the name **define** is pre-empted. There is no provision for automatic variables other than parameters. Examples:  
define f()  
define f(a, b, c)
- All labels except **define** (even **end**), must have a nonempty statement.
- Labels, functions, and variables must all have distinct names. In particular, the nonempty statement on **end** cannot merely name a label.
- If **start** is a label in the program, program execution begins there. If not, execution begins with the first executable statement. **define** is not an executable statement.
- There are no built-in functions.

- Parentheses for arithmetic are not needed. Normal precedence applies. Because of this, the arithmetic operators \ (backslash) and \* (asterisk) must be set off by spaces.
- The right side of assignments must be nonempty.
- Either ' (single quotation mark) or " (double quotation mark) can be used for literal quotation marks.
- The pseudo-variable **syspvt** is not available.

## Related Information

The following command: “**awk**” on page 81.

# sort

---

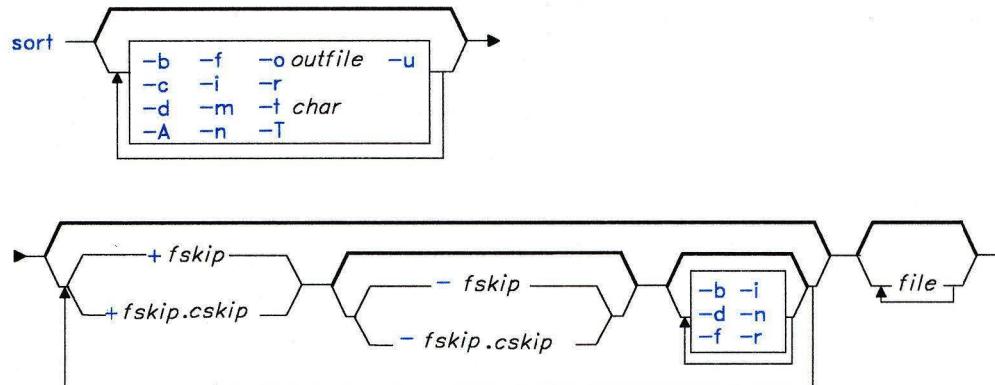
## sort

---

### Purpose

Sorts or merges files.

### Syntax



OL805380

### Description

The **sort** command sorts lines in its input *files* and writes the result to standard output. It treats all of its input *files* as one file when it performs the sort. A - (minus) in place of a file name specifies standard input. If you do not specify any file names, it sorts standard input.

The default sort key (the part of the line used for sorting) is an entire line. Default ordering is lexicographic by characters in the collating sequence. The file `/usr/pub/ascii` shows the default collating sequence. To change the default collating sequence, see “**ctab**” on page 257.

The two numbers, *fskip* and *cskip*, specify the sort key. Both numbers have two parts, as follows:

`+fskip.cskip`  
`-fskip.cskip`

The *fskip* specifies the number of fields to skip from the beginning of the input line, and *cskip* specifies the number of additional characters to skip to the right beyond that point. For both the starting point (*+fskip.cskip*) and the ending point (*-fskip.cskip*) of a sort key, *fskip* is measured from the beginning of the input line, and *cskip* is measured from the last field skipped. If you omit *.cskip*, .0 is assumed. If you omit *fskip*, 0 is assumed. If you omit the ending field specifier (*-fskip.cskip*), the end of the line is the end of the sort key.

You can supply more than one sort key by repeating *+fskip.cskip* and *-fskip.cskip*. In cases where you specify more than one sort key, keys specified further to the right on the command line are compared only after all earlier keys are sorted. For example, if the first key is to be sorted in numerical order and the second in dictionary order, all strings that start with the number one are sorted alphabetically before the strings that start with the number two. Lines that are identical in all keys are sorted with all characters significant. You can also specify different flags for different sort keys in multiple sort keys. See the examples for illustration.

A field is one or more characters bounded by the beginning of a line and the current field separator, or one or more characters bounded by a the field separator on either side. The space character is the default field separator.

#### Notes:

1. Lines longer than 1024 are truncated.
2. The maximum number of fields on a line is 10.
3. The **sort** command will not process files containing imbedded commands.

## Flags

- A Sorts on a byte-by-byte basis. This sort is functionally compatible with the Version 1.1 **sort** command, prior to the addition of international character support.
- b Ignores leading blanks, spaces, and tabs in sort key comparisons.
- c Checks that the input is sorted according to the ordering rules specified in the flags. Displays nothing unless the file is not sorted.
- d Sorts in dictionary order. Only letters, digits and blanks are considered in comparisons.
- f Merges uppercase and lowercase letters. Case is not considered in the sorting, so that initial-capital words and all-capital words are not grouped together at the beginning of the output.

## sort

---

**-i** Sorts only by characters in the ASCII range octal 040-0176 (all printable characters and the space character) in non-numeric comparisons.

---

### Japanese Language Support Information

Sorts only by printable characters in non-numeric comparisons.

---

**-m** Merges only; the input is already sorted.

**-n** Sorts any initial numeric strings (consisting of optional blanks, optional minus signs, and zero or more digits with optional decimal point) by arithmetic value. The **-n** flag automatically gives you the **-b** flag.

**-o** *outfile* Directs output to *outfile* instead of standard output. *outfile* can be the same as one of the input *files*.

**-r** Reverses the order of the specified sort.

**-tchar** Sets field separator character to *char*. To specify the tab character as the field separator, you must enclose it in single quotation marks (' ').

**-T** Uses current directory instead of default directory for temporary files.

**-u** Suppresses all but one in each set of equal lines. Ignored characters (such as leading tabs and spaces) and characters outside of sort keys are not considered in this type of comparison.

## Examples

1. To perform a simple sort:

```
sort fruits
```

This displays the contents of `fruits` sorted in ascending lexicographic order. This means that the characters in each column are compared one by one, including spaces, digits, and special characters. For instance, if `fruits` contains the text:

```
banana
orange
Persimmon
apple
%%banana
apple
ORANGE
```

then **sort** displays:

```
%%banana
ORANGE
Persimmon
apple
apple
banana
orange
```

This order follows from the fact that in the ASCII collating sequence, % (percent sign) precedes the uppercase letters, which precede the lowercase letters. If the system uses a character set other than ASCII, your results may be different.

2. To sort in dictionary order:

```
sort -d fruits
```

This sorts and displays the contents of `fruits`, comparing only letters, digits, and blanks. If `fruits` is the same as in Example 1, then **sort** displays:

```
ORANGE
Persimmon
apple
apple
%%banana
banana
orange
```

The `-d` flag tells **sort** to ignore the % character because it is not a letter, digit, or blank. This puts %%banana next to banana.

3. To group lines that contain uppercase and special characters with similar lowercase lines:

```
sort -d -f fruits
```

This ignores special characters (`-d`) and differences in case (`-f`). Given the `fruits` of Example 1, this displays:

```
apple
apple
%%banana
banana
ORANGE
orange
Persimmon
```

4. To sort as in Example 3 and remove duplicate lines:

```
sort -d -f -u fruits
```

The `-u` flag tells **sort** to remove duplicate lines, making each line of the file unique. This displays:

```
apple
%%banana
orange
Persimmon
```

Note that not only was the duplicate `apple` removed, but `banana` and `ORANGE` as well. These were removed because the `-d` told **sort** to treat `%%banana` as if it were `banana`, and the `-f` told it to treat `ORANGE` as `orange`. Thus, **sort** considered `%%banana` to be a duplicate of `banana` and `ORANGE` a duplicate of `orange`.

**Note:** There is no way to predict which duplicate lines `sort -u` will keep and which it will remove.

5. To sort as in Example 3 and remove duplicates, unless capitalized or punctuated differently:

```
sort -u +0 -d -f +0 fruits
```

The `+0 -d -f` does the same type of sort done with `-d -f` in Example 3. Then the `+0` performs another comparison to distinguish lines that are not actually identical. This prevents `-u` from removing them.

Given the `fruits` file shown in Example 1, the added `+0` distinguishes `%%banana` from `banana` and `ORANGE` from `orange`. However, the two instances of `apple` are identical, so one of them is deleted.

```
apple
%%banana
banana
ORANGE
orange
Persimmon
```

6. To specify the character that separates fields:

```
sort -t: +1 vegetables
```

This sorts `vegetables`, comparing the text that follows the first colon on each line. The `+1` tells **sort** to ignore the first field and to compare from the start of the second field to the end of the line. The `-t:` tells **sort** that colons separate fields. If `vegetables` contains:

```
yams:104
turnips:8
potatoes:15
carrots:104
green beans:32
radishes:5
lettuce:15
```

then **sort** displays:

```
carrots:104
yams:104
lettuce:15
potatoes:15
green beans:32
radishes:5
turnips:8
```

Note that the numbers are not in numeric order. This happened because a lexicographic sort compares each character from left to right. In other words, “3” comes before “5” and “2” comes before “ ”, so “32” comes before “5 ”.

7. To sort numbers:

```
sort -t: +1 -n vegetables
```

This sorts `vegetables` numerically on the second field. If `vegetables` is the same as in Example 6, then **sort** displays:

```
radishes:5
turnips:8
lettuce:15
potatoes:15
green beans:32
carrots:104
yams:104
```

8. To sort on more than one field:

```
sort -t: +1 -2 -n +0 -1 -r vegetables
```

This performs a numeric sort on the second field (+1 -2 -n). Within that ordering, it sorts the first field in reverse alphabetic order (+0 -1 -r). The output looks like this:

## sort

---

```
radishes:5  
turnips:8  
potatoes:15  
lettuce:15  
green beans:32  
yams:104  
carrots:104
```

Now the lines are sorted in numeric order. When two lines have the same number, they appear in reverse alphabetic order.

9. To replace the original file with the sorted text:

```
sort -o vegetables vegetables
```

This stores the sorted output into the file `vegetables` (`-o vegetables`).

## Files

`sort.c`     Contains sort definitions.

## Related Information

The following commands: “**comm**” on page 183, “**join**” on page 547, and “**uniq**” on page 1118.

The “Overview of International Character Support” in *Managing the AIX Operating System*.

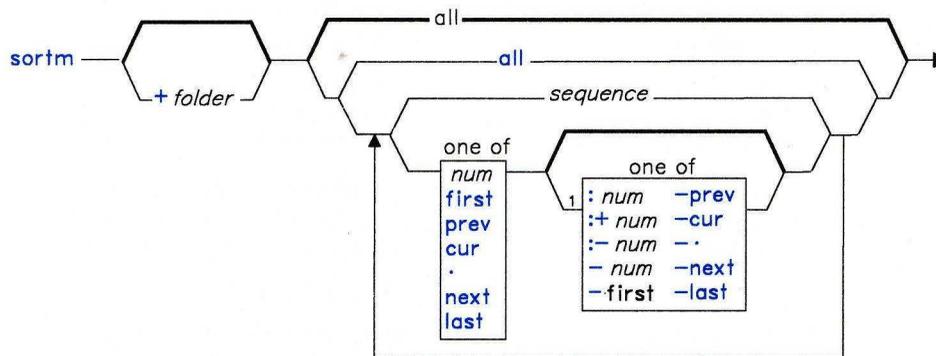
The discussion of Japanese Language Support in *Japanese Language Support User’s Guide*.

# sortm

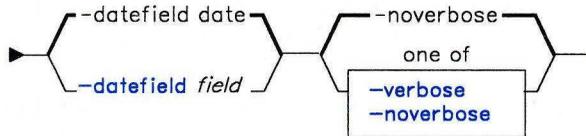
## Purpose

Sorts messages.

## Syntax



AJ2FL209



`sortm -help`

AJ2FL211

<sup>1</sup> Do not put a blank between these items.

OL805308

## Description

The `sortm` command is used to sort messages. The `sortm` command is part of the MH (Message Handling) package and can be used with MH and AIX commands.

## sortm

---

The **sortm** command sorts the messages according to the date in a header field. By default, **sortm** parses the **Date:**, but you can use the **-datefield** flag to specify another field. **sortm** numbers the sorted messages consecutively beginning with 1 (one). Messages that are in the folder, but not specified to be sorted, are placed after the sorted messages. **sortm** displays a message if it cannot parse a date field.

### Flags

**-datefield** *field* Specifies the header field to be used in the sort. The default field is **Date:**.

**+folder** *msgs* Specifies the messages to be sorted. You can use the following message references when specifying *msgs*:

<i>num</i>	<b>first</b>	<b>prev</b>
<b>cur</b>	.	<b>next</b>
<b>last</b>	<b>all</b>	<i>sequence</i>

The default is all messages in the current folder. If you specify a folder, it becomes the current folder. The current message remains the current message, even if it moves during the sort.

**-help** Displays help information for the command.

**-noverbose** Does not display information during the sort. This flag is the default.

**-verbose** Displays information during the sort. This information allows you to monitor the steps involved.

### Profile Entries

**Current-Folder:** Sets your default current folder.  
**Path:** Specifies your *user\_mh\_directory*.

### Files

`$HOME/.mh-profile` The MH user profile.

### Related Information

The MH command “**folder**” on page 429.

The **mh-profile** file in *AIX Operating System Technical Reference*.

The “Overview of the Message Handling Package” in *Managing the AIX Operating System*.

---

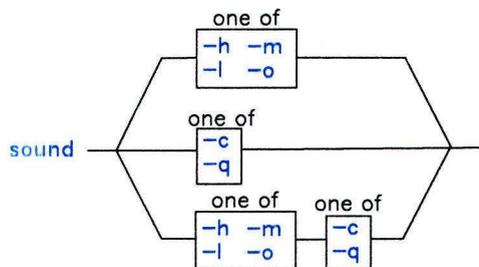
# sound

---

## Purpose

Controls the volume and click of the keyboard speaker.

## Syntax



OL805415

## Description

The **sound** command controls the volume of the sound output (the console bell and the keyboard click) and, additionally, whether or not the keyboard click is produced. You can modify these two sound characteristics independently of each other.

The system startup process sets the sound volume to medium.

**Note:** You can run **sound** only from the console (**/dev/console**).

## Flags

You must select at least one flag from the following two groups of flags or, optionally, one flag from each of the two groups. The first group of flags controls the volume of all sound output:

- h** Sets the volume to high.
- l** Sets the volume to low.
- m** Sets the volume to medium.
- o** Turns the volume off.

## sound

---

The second group of flags controls whether or not click sounds are produced:

- c Turns clicking on.
- q Turns clicking off (quiet).

### Example

To set the volume to low and turn the click function on:

```
sound -lc
```

In addition to turning on the keyboard click (-c), this command sets the volume of both the bell and the click to low (-l).

---

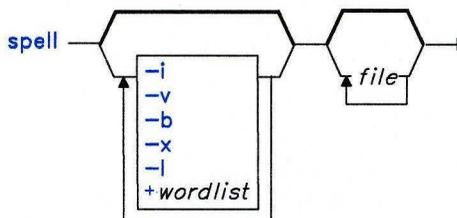
# spell

---

## Purpose

Finds spelling errors.

## Syntax



`/usr/lib/spell/hashmake`  $\rightarrow$

`/usr/lib/spell/spellin`  $\text{--- num} \rightarrow$

`/usr/lib/spell/hashcheck`  $\text{--- spellinglist} \rightarrow$

OL805304

## Description

The **spell** command reads words in *file* and compares them to those in a spelling list. Words that cannot be matched in the spelling list or derived from words in the spelling list (by applying certain inflections, prefixes, and/or suffixes) are written to standard output. If you do not specify a file to read, **spell** reads standard input.

The **spell** command ignores the same **troff**, **tbl**, and **eqn** constructs as the **deroff** command.

The coverage of the spelling list is uneven. You should create your own dictionary of special words used in your files.

Certain auxiliary files can be specified by file name parameters; see “Files” on page 971. Copies of all output are accumulated in the history file.

Three routines help maintain and check the hash lists used by **spell**.

## spell

---

<code>/usr/lib/spell/hashmake</code>	Reads a list of words from standard input and writes the corresponding nine-digit hash code to standard output.
<code>/usr/lib/spell/spellin num</code>	Reads <i>num</i> hash codes from standard input and writes a compressed spelling list to standard output.
<code>/usr/lib/spell/hashcheck spellinglist</code>	Reads a compressed <i>spellinglist</i> and recreates the nine-digit hash codes for all the words in it; it writes these codes to standard output.

## Flags

<code>-b</code>	Checks British spelling.
<code>-i</code>	Suppresses processing of included files.
<code>-l</code>	Follows the chain of all included files ( <code>.so</code> and <code>.nx</code> formatting commands). Without this flag, <code>spell</code> follows chains of all included files except for those beginning with <code>/usr/lib</code> .
<code>-v</code>	Displays all words not literally in the spelling list and indicates plausible derivations from the words.
<code>-x</code>	Displays every plausible word stem with an = (equal sign).
<code>+ wordlist</code>	Checks <i>wordlist</i> for additional word spellings. <i>wordlist</i> is the name of a file you provide that contains a sorted list of words, one per line. With this flag, you can specify a set of correctly spelled words (in addition to <code>spell</code> 's own spelling list) for each job.

## Examples

1. To check your spelling:

```
spell chap1 >mistakes
```

This creates a file named `mistakes` containing all the words found in `chap1` that are not in the system spelling dictionary. Some of these may be correctly spelled words that `spell` does not know about. It is a good idea to save the output of `spell` in a file because the word list may be long.

2. To check British spelling:

```
spell -b chap1 >mistakes
```

This checks `chap1` against the British dictionary and writes the questionable words in `mistakes`.

3. To see how **spell** derives words:

```
spell -v chap1 >deriv
```

This lists the words that are not found literally in the dictionary, but are derived forms of dictionary words. The prefixes and suffixes used to form the derivative are indicated for each word. Words that do not appear in the dictionary at all are also listed.

4. To check your spelling against an additional word list:

```
spell +newwords chap1
```

This checks the spelling of words in `chap1` against the system dictionary and against `newwords`. The file `newwords` lists words in alphabetical order, one per line. You can create this file with a text editor, such as **ed**, and alphabetize it with the **sort** command.

## Files

D_SPELL = /usr/lib/spell/hlist[ab]	Hashed spelling lists, American and British.
S_SPELL = /usr/lib/spell/hstop	Hashed stop list.
H_SPELL = /usr/lib/spell/spellhist	History file.
/usr/lib/compress	Executable shell program to compress the history file.
/usr/lib/spell/spellprog	Program.

## Related Information

The following commands: “**deroff**” on page 313, “**eqn, neqn, checked**” on page 395, “**sed**” on page 887, “**sort**” on page 958, “**tbl**” on page 1053, “**tee**” on page 1060, and “**troff**” on page 710.

# spline

---

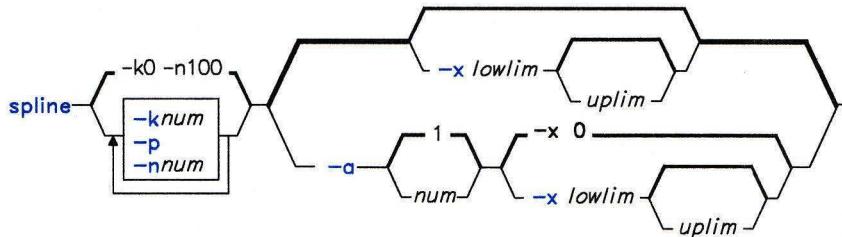
## spline

---

### Purpose

Interpolates smooth curve.

### Syntax



OL805261

### Description

The **spline** command reads from the standard input pairs of numbers that represent the coordinates of a point on an x,y axis. From this input, **spline** calculates the coordinates of points to form a smooth curve through the points in the input set. It then writes these points to standard output. The output points are approximately equally spaced and includes the points that you provided as input. The cubic spline output has two continuous derivatives, and enough points so that when plotted with the **graph** command it looks smooth.

When data is not strictly monotone in x, **spline** reproduces the input without interpolating extra points.

You can only use 1,000 input points.

### Flags

**-anum** Supplies abscissas automatically; spacing is given by the next parameter or is assumed to be 1 if the next parameter is not a number.

**-knum** Uses the constant *num* in the boundary value calculation:

$$y_0 = ky_1, y = numy$$

The default for *num* is zero.

- num*** Spaces output points so that approximately *num* intervals occur between the lower and upper *x* limits (set with the **-x** flag). The default *num* is 100.
- p** Makes output periodic, that is, matches derivatives at ends. First and last input values should normally agree.
- x*lowlim*[*uplim*]** Sets lower and upper *x* limits as *lowlim* and *uplim*. Normally, these limits are calculated from the data. Automatic abscissas start at lower limit, defaults to zero.

## Related Information

The following command: “**graph**” on page 494.

# split

---

# split

---

## Purpose

Splits a file into pieces.

## Syntax



OL805262

## Description

The **split** command reads *file* and writes it in *num*-line pieces (default 1000 lines) to a set of output files. The name of the first output file is *prefixaa*, the second is *prefixab*, and so on lexicographically, through *prefixzz* (a maximum of 676 files). *prefix* cannot be longer than 12 characters. If you do not specify an output name, *x* is assumed.

If you do not specify an input file, or if you specify - (minus) in place of *file*, then **split** reads standard input.

## Examples

1. To split a file into 1000-line segments:

```
split book
```

This splits `book` into 1000-line segments named `xaa`, `xab`, `xac`, and so forth.

2. To split a file into 50-line segments and specify the file name prefix:

```
split -50 book sect
```

This splits `book` into 50-line segments named `sectaa`, `sectab`, `sectac`, and so forth.

## Related Information

The following commands: “**bfs**” on page 110 and “**csplit**” on page 252.

---

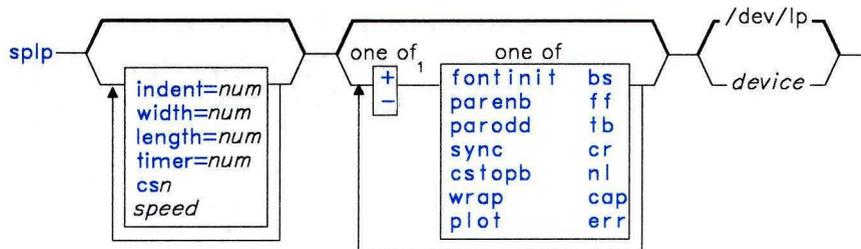
# splp

---

## Purpose

Changes or displays printer driver settings.

## Syntax



OL805263

<sup>1</sup> Do not put a blank between these items.

OL805308

## Description

The **splp** command changes or displays settings for a printer driver (*device*). The default *device* is **/dev/lp0**. If you do not specify any flags, **splp** reports the current settings for the specified *device*. Select flags to change the current settings. No other processing is done, and there is no other output.

The changes that **splp** makes remain in effect until the next time you restart the system or rerun **splp**. You can run **splp** from the **/etc/rc** command file to configure your printer each time you start up the system.

**Note:** When the **print** command is used with the backend **piobe**, **splp** is set as **+plot**. Any parameters set by the user are ignored. If a file is redirected using the **cat** command instead of the **print** command, the **splp** settings are active.

---

### Japanese Language Support Information

This command has not been modified to support Japanese characters.

---

## Flags

- +ascii** **-ascii**(  
Selects ASCII **)PostScript**( mode for the Personal Proprinter Adapter.
- timer = num** Sets the time out period to *num* seconds, where *num* is an integer.
- indent = num** Indents *num* columns, where *num* is an integer.
- length = num** Prints *num* lines per page, where *num* is an integer.
- width = num** Prints *num* columns, where *num* is an integer
- +bs** **(-bs)** Sends (does not send) backspaces to the printer.
- +cap** **(-cap)** Converts (does not convert) all lowercase characters to uppercase.
- +cr** **(-cr)** Sends carriage returns (translates carriage returns to carriage return - line feeds).
- +cstopb** **(-cstopb)**  
Selects 2 (1) stop bits per character.
- cs5 cs6 cs7 cs8**  
Selects character size. See **termio** in *AIX Operating System Technical Reference* for additional information on character size.
- +err** **(-err)** Issues (does not issue) a signal (SIGIOINT) upon receiving a VRM error and attempts to resume I/O.
- +ff** **(-ff)** Sends form feeds (simulates a form feed with line feeds or carriage returns).
- +fontinit** **(-fontinit)**  
Indicates that fonts are (are not) loaded. Use this flag to control the initialization of fonts for the 3812 Pageprinter or the Personal Proprinter Adapter.
- +nl** **(-nl)** Sends line feeds (translates line feeds to line feed - carriage returns).
- +parenb** **(-parenb)**  
Enables (disables) parity generation and detection.
- +parodd** **(-parodd)**  
Selects odd (even) parity.
- +plot** Sends all characters to the printer unmodified. This overrides other settings.
- plot** Translates characters according to the settings.
- +sync** **(-sync)**  
Does not (does) return immediately without waiting for all data to be sent out.

- 
- +tb (-tb)** Expands (does not expand) tabs on eight position boundaries.
- +wrap (-wrap)** Wraps (truncates) characters beyond the specified **width** to the next line and (with **+wrap**), prints " . . ." before the new-line character.
- 50 75 110 134 150 300 600 1200 1800 2400 4800 9600 exta extb**  
Sets the *speed* to the specified number of bits per second (**exta** is 19200).

## Examples

1. To display the current printer settings:

```
splp
```

2. To change the printer settings:

```
splp width=80 +wrap +cap
```

This changes the settings of the **/dev/lp** printer for 80-column paper (**width = 80**). It wraps each line that is more than 80 columns wide onto a second line (**+wrap**), and prints all alphabetic characters in uppercase (**+cap**).

## Related Information

The following command: “**lp**” on page 593.

The **lp** file in *AIX Operating System Technical Reference*.

# spost

---

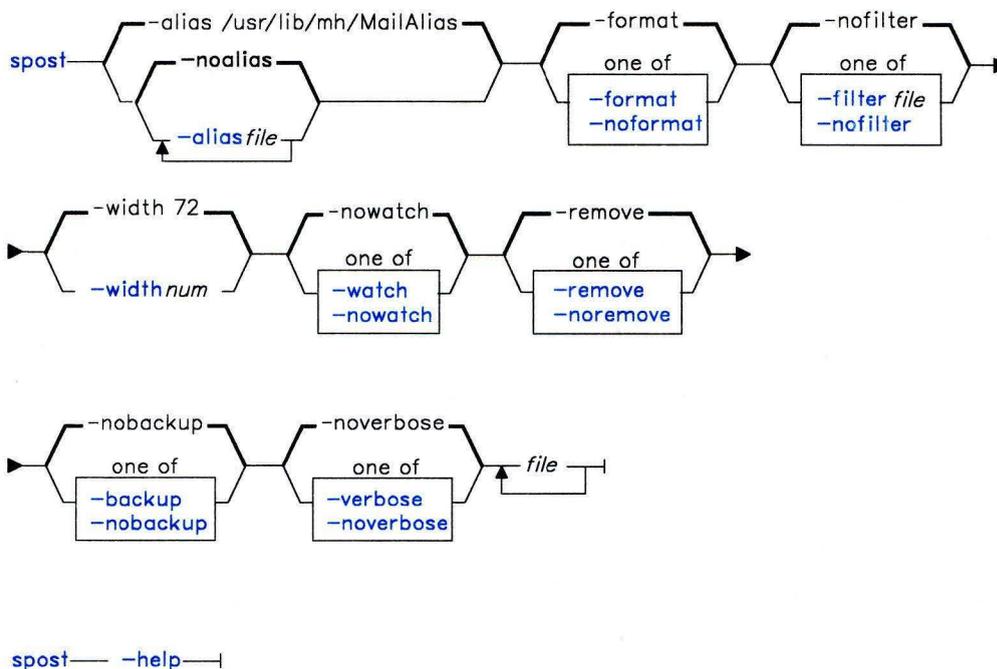
## spost

---

### Purpose

Delivers a message.

### Syntax



AJ2FL251

AJ2FL252

### Description

The **spost** command is used to route messages to the proper destinations. **spost** is not designed to be run directly by the user; it is designed to be called by other programs. The **spost** command is part of the MH (Message Handling) package.

The **spost** command searches all components of a message that specify a recipient's address and parses each address to check for proper format. **spost** puts proper addresses in the standard format and invokes the **sendmail** command. **spost** perform a similar function as the **post** command, but **spost** does less internal error checking than **post**.

## Flags

- alias Files** Searches the specified mail alias file for addresses. You can use this flag repetitively to specify multiple mail alias files. **spost** automatically searches the file `/usr/lib/mh/MailAliases`.
- backup** Renames the message file by placing a `,` (comma) before the file name after **spost** successfully posts the message.
- filter Files** Uses the header components in the specified file for copies of the message sent to **Bcc:** recipients.
- format** Puts all recipient addresses in a standard format for the delivery transport system. This flag is the default.
- help** Displays help information for the command.
- nofilter** Strips the **Bcc:** header from the message and sends it to recipients specified in the **Bcc:** component. This flag is the default.
- noalias** Does not use any alias files for delivering the message.
- nobackup** Does not rename the message after posting the file. This flag is the default.
- noformat** Does not alter the format of the recipient addresses.
- noremove** Does not remove the temporary message file after posting the message.
- noverbose** Does not display information during the delivery of the message to the **sendmail** command. This flag is the default.
- nowatch** Does not display information during delivery by the **sendmail** command. This flag is the default.
- remove** Removes the temporary message file after the successful completion of posting the message. This flag is the default.
- verbose** Displays information during the delivery of the message to the **sendmail** command. This information allows you to monitor the steps involved.
- watch** Displays information during the delivery of the message by the **sendmail** command. This information allows you to monitor the steps involved.
- width *num*** Sets the width of components that contain addresses. The default is 72 columns.

## Files

`$HOME/.mh-profile`  
`/temp/pstnum`

The MH user profile.  
The temporary message file.

## Related Information

Other commands: “**ali**” on page 48, “**conflict**” on page 196, “**mhmail**” on page 646, “**post**” on page 758, “**send**” on page 893, “**sendmail**” on page 897, and “**whom**” on page 1222.

The **mh-alias**, **mh-format**, **mh-mail**, and **mh-profile** files in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

---

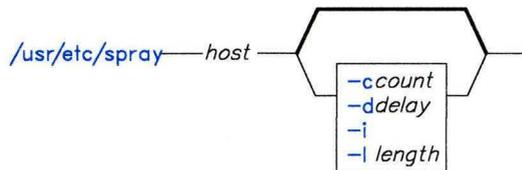
# spray

---

## Purpose

Sends specified number of packets to host when NFS is installed.

## Syntax



OL805509

## Description

The **spray** command sends a one-way stream of packets to *host* using RPC. It reports how many packets were received, and at what transfer rate. The *host* parameter can be a network address or a name.

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

## Flags

- c** Specifies the number of packets to send. The default is the number of packets required to make the total stream size 100,000 bytes.
- d** Specifies the time, in microseconds, the system pauses between sending each packet. The default is 0.
- i** Specifies ICMP echo packets rather than RPC. Since ICMP echoes automatically, it creates a two-way stream.
- l** Number of bytes in the Ethernet packet that holds the RPC call message. The default is 86 bytes, which is the length of RPC and UDP/IP headers.  
The data is encoded using XDR. Since XDR deals only with 32-bit quantities, **spray** rounds smaller *length* values up to the nearest 32-bit value.

## spray

---

**Note:** If the **length** is greater than 1514, the RPC call does not fit into one Ethernet packet, and the **length** field will not have a simple correspondence to the Ethernet packet size.

### Related Information

The following command: “**sprayd**” on page 983.

---

# sprayd

---

## Purpose

Receives packets sent by the **spray** command

## Syntax

`/usr/etc/rpc.sprayd`—|

OL805510

## Description

The **sprayd** daemon receives the packets sent when the **spray** command is issued. The **inetd** daemon invokes **sprayd**.

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

## File

`/etc/inetd.conf` TCP/IP configuration file.

## Related Information

The following command: “**spray**” on page 981.

### Purpose

Provides tools for analyzing numerical data.

### Description

The **stat** commands, residing in **/usr/bin/graf**, provide a package of tools for analyzing data. All numerical data are stored in vectors. A **vector** is a sequence of numbers separated by delimiters, where a number has the form:

$$[sign](digits)(.digits)[e[sign]digits]$$

Fields surrounded by brackets are optional; one or both of those surrounded by parentheses are required. Any input character that is not part of a number is assumed to be a delimiter.

Vectors are text strings that can be stored in text files and created and modified by text editors.

**Note:** Some commands limit the size of an input vector.

These commands can be divided into four classes:

- Those that produce an output vector based upon definable parameters (generators).
- Those that operate upon an input vector and output the resulting value (transformers).
- Those that perform mathematical or statistical operations on vectors (summarizers).
- Those that convert vectors into a format that can be viewed pictorially (translators).

The following parameters are used to designate the expected type of the value:

*c*      A character value.  
*i*      An integer value.  
*f*      A floating-point or integer value.  
*file*    A file name.

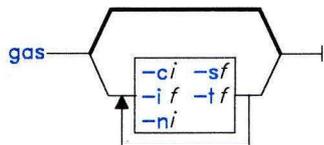
*vector* A vector taken from standard input or the name of a file containing a vector. Except for the **gas**, **prime**, and **rand** commands, all of the commands discussed under **stat** read vectors from standard input (by default) or from text files as specified on the command line. A file name of - (minus) specifies standard input in a file list.

*string* A character string (quoted if it includes white space).

## Commands That Produce Definable Vectors (Generators)

### gas

#### Syntax



OL805513

#### Description

The **gas** command produces an additive sequence.

#### Flags

- ci** Specifies the number of columns per line of output (5 by default).
- if** Specifies the increment between successive elements (1 by default).
- ni** Specifies the number of elements in the vector (10 by default).
- sf** Specifies the starting point of the sequence (1 by default).
- tf** Specifies the end of the sequence (infinity by default).

#### Examples

1. To generate the numbers 1 through 10:  
`gas`
2. To generate the sequence .01 .02 .03 .04 .05:

## stat

---

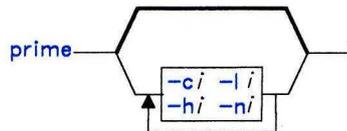
```
gas -s.01,t.05,i.01
```

3. To generate the sequence 3 5 3 5:

```
gas -s3,t5,i2,n4
```

## prime

### *Syntax*



OL805514

### *Description*

The **prime** command generates consecutive prime numbers.

### *Flags*

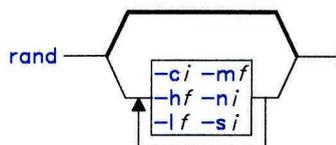
- ci Specifies the number of columns per line of output (5 by default).
- hi Specifies the high boundary (infinity by default).
- li Specifies the low boundary (2 by default).
- ni Specifies the number of elements in the sequence (10 by default).

### *Example*

To generate all prime numbers between 200 and 300:

```
prime -l200,-h300
```

---

**rand****Syntax**

OL805515

**Description**

The **rand** command generates a random sequence of numbers.

**Flags**

- ci** Specifies the number of columns per line of output (5 by default).
- hf** Specifies the high boundary (1 by default).
- lf** Specifies the low boundary (0 by default).
- mf** Specifies the high boundary where  $hf = mf + lf$ .
- ni** Specifies the number of elements in the sequence (10 by default).
- si** Specifies a seed number (1 by default).

**Example**

To produce a random sequence:

```
rand
```

This generates ten random numbers between 0 and 1.

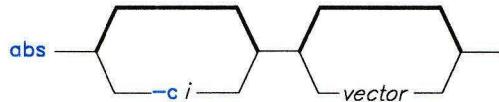
```
rand -l10,m25,c3
```

This generates ten random numbers between 10 and 35, three per line.

## Commands That Map Input to Output (Transformers)

### abs

#### *Syntax*



OL805516

#### *Description*

The **abs** command provides the absolute value of a number.

#### *Flag*

**-ci** Specifies the number of columns per line of output (5 by default).

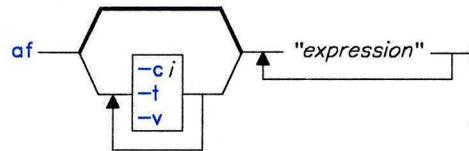
#### *Example*

To obtain the absolute value of each element in a vector:

```
abs -c3 myfile
```

This produces the absolute value of each number in the file `myfile` and displays these values three per line.

af

**Syntax**

OL805517

**Description**

The **af** command performs arithmetic operations on numbers.

**Expressions**

Expression operands are:

**Vectors** File names with the restriction that they must begin with a letter and be composed only of letters, digits, and the `_` (underscore) and `.` (dot) characters. The first unknown file name (one not in the current directory) references standard input.

**Functions** The name of a command followed by the command arguments in parentheses. List arguments as you would on the command line.

**Constants** Floating-point and integer numbers (but not E notation).

Expression operators are, in order of decreasing precedence:

`'v` The next value from vector *v*.

`x^y -x` The value *x* raised to the power *y*; the negation of *x*. Both associate right to left.

`x*y x/y x%y` The value *x* multiplied by, divided by, modulo *y*, respectively. All associate left to right.

`x+y x-y` The value *x* plus or minus *y*. Both associate left to right.

`x,y` The value of *x* followed by the value of *y*. This associates from left to right.

You can use parentheses to alter precedence. Because many of the operator characters are special to the shell, it is good practice to quote expression arguments.

### *Flags*

- ci* Specifies the number of columns per line of output (5 by default).
- t* Cause the output to be titled from the vector on standard input.
- v* Echoes function expansions.

### *Examples*

1. To perform arithmetic operations:

```
af "3+4*5"
```

This yields 23.

2. To produce a matrix:

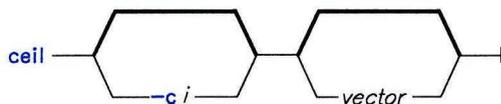
```
af "A, 'A,A+'A, B"
```

This yields a four-column matrix with columns of:

- a. odd elements from vector A
  - b. even elements from A
  - c. sum of adjacent odd and even elements from A
  - d. elements from vector B.
3. To use functions:

```
af "sin (A)^2"
```

This yields the square of the sin of the elements of vector A.

**ceil****Syntax**

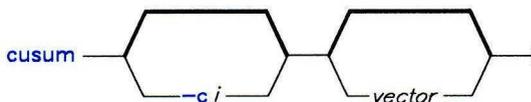
OL805518

**Description**

The `ceil` command rounds a number up to the next integer.

**Flag**

`-ci` Specifies the number of columns per line of output (5 by default).

**cusum****Syntax**

OL805519

**Description**

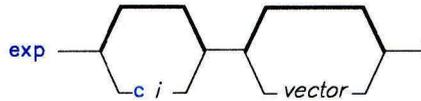
The `cusum` command calculates a cumulative sum. Output is a vector with the  $i$ th element being the sum of the first  $i$  elements from the input vector.

**Flag**

`-ci` Specifies the number of columns per line of output (5 by default).

## exp

### *Syntax*



OL805549

### *Description*

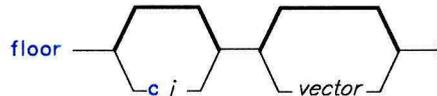
The **exp** command provides the exponential function. Output is a vector with elements  $e$  raised to the  $x$  power, where  $e$  is approximately 2.71828 and  $x$  is each element in the input vector.

### *Flag*

**-ci** Specifies the number of columns per line of output (5 by default).

## floor

### *Syntax*



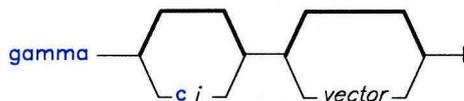
OL805550

### *Description*

The **floor** command rounds a number down to the nearest integer.

### *Flag*

**-ci** Specifies the number of columns per line of output (5 by default).

**gamma****Syntax**

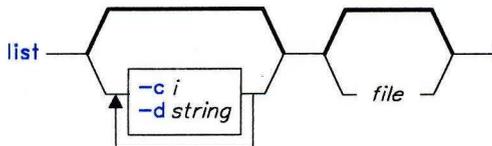
OL805551

**Description**

The **gamma** command provides the gamma function.

**Flag**

**-ci** Specifies the number of columns per line of output (5 by default).

**list****Syntax**

OL805552

**Description**

The **list** command lists vector elements.

**Flags**

**-ci** Specifies the number of columns per line of output (5 by default).

**-dstring** Specifies delimiter characters. If you do not specify **-d**, any character that is not part of a number is considered a delimiter. If you specify **-d**, the space, tab, and new-line characters, plus the characters in *string* are delimiters.

Only numbers surrounded by delimiters are listed.

**Examples**

1. To output each element:

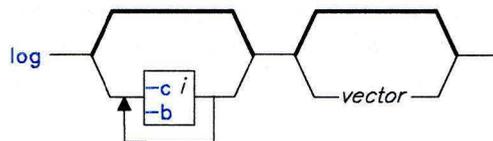
```
list -c3 myfile
```

This outputs each element in `myfile`, three per line.

2. To specify delimiters:

```
list -d\\, myfile
```

This outputs each element of `myfile` that is delimited by commas or white space, five per line. A comma requires two backslashes because it is a special character for `list`.

**log****Syntax**

OL805520

**Description**

The `log` command provides the logarithmic function.

**Flag**

`-ci` Specifies the number of columns per line of output (5 by default).

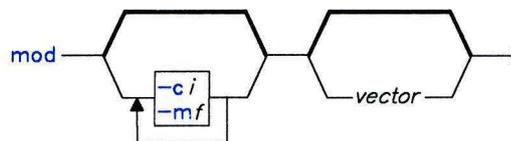
`-bf` Specifies the base (e by default).

**Example**

To calculate a logarithm:

```
log -b2,c3 mydata
```

This outputs the logarithm base 2 of each element in `mydata`, three per line.

**mod*****Syntax***

OL805521

***Description***

The **mod** command returns the modulo. The output is a vector with each element being the remainder of dividing the corresponding element from the input vector by the modulus.

**-ci** Specifies the number of columns per line of output (5 by default).

**-mf** Specifies the modulus (2 by default).

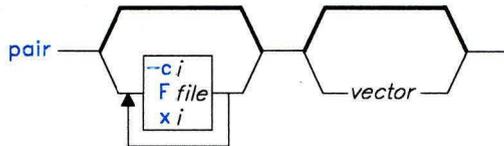
***Example***

To output remainders:

```
mod -m8,c3 mydata
```

This outputs the elements of `mydata` modulo 8, three per line.

## pair

**Syntax**

OL805522

**Description**

The **pair** command pairs elements. Output is a vector with elements taken alternatively from a base vector and from another input vector.

**Flags**

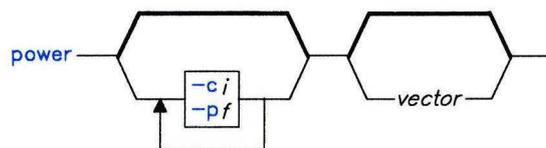
- ci** Specifies the number of columns per line of output (5 by default).
- Ffile** The file containing the base vector. If you do not specify **-F**, then the base vector comes from standard input. If both the base vector and the paired vector come from standard input, the base vector precedes the paired vector.
- xi** The number of elements per group in the base vector (1 by default).

**Example**

To pair elements:

```
pair -x3,Fbasefile datafile
```

This outputs a vector with three elements from basefile, one from datafile, three from basefile, one from datafile, and so on.

**power****Syntax**

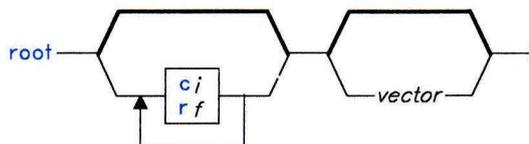
OL805523

**Description**

The **power** command raises a number to a power.

**Flag**

- ci Specifies the number of columns per line of output (5 by default).
- pf Specifies the power (2 by default).

**root****Syntax**

OL805524

**Description**

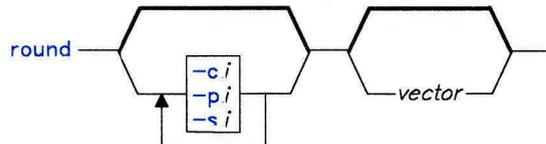
The **root** command takes the root of a number.

**Flags**

- ci Specifies the number of columns per line of output (5 by default).
- rf Specifies the root (2 by default).

## round

### *Syntax*



OL805525

### *Description*

The **round** command rounds a number to the nearest integer (.5 rounds to 1).

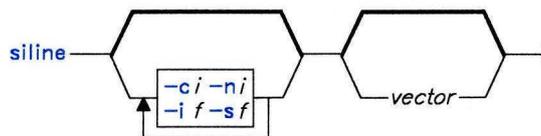
### *Flags*

- ci* Specifies the number of columns per line of output (5 by default).
- pi* Specifies the number of places after the decimal point (0 by default).
- si* Specifies the number of significant digits.

### *Example*

To round numbers to two significant digits:

```
round -s2,c3 mydata
```

**siline*****Syntax***

OL805526

***Description***

The **siline** command generates a line, given slope and intercept.

***Flags***

- ci** Specifies the number of columns per line of output (5 by default).
- if** Specifies the intercept (0 by default).
- ni** Specifies the number of positive integers.
- sf** Specifies the slope of the line.

***Example***

To output a linear fit:

```
siline -\lref -o,FA B` A
```

This outputs a simple linear fit of vector B on vector A (The **o** flag of **lreg** outputs the slope and intercept in option form of B regressed on A.)

## sin

### Syntax



OL805527

### Description

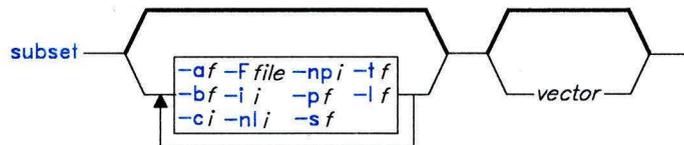
The **sin** command provides the sine function.

### Flags

**-ci** Specifies the number of columns per line of output (5 by default).

## subset

### Syntax



OL805528

### Description

The **subset** command produces a subset of the numbers in a vector.

### Flags

**-af** Specifies the number above which subset members are selected.

**-bf** Specifies the number below which subset members are selected.

**-ci** Specifies the number of columns per line of output (5 by default).

**-Ffile** Specifies the file containing the master vector.

**-ii** Specifies the increment between successive elements (1 by default).

- lf** The number of elements to leave.
- nli** Specifies that the master file contains element numbers to leave.
- npi** Specifies that the master file contains element numbers to pick.
- pf** The number of elements to pick.
- sf** Specifies the starting point of the sequence (1 by default).
- tf** Specifies the end of the sequence (32,767 by default).

### Examples

1. To specify the even elements of a vector:

```
subset -i2,s2 myfile
```

2. To specify corresponding elements:

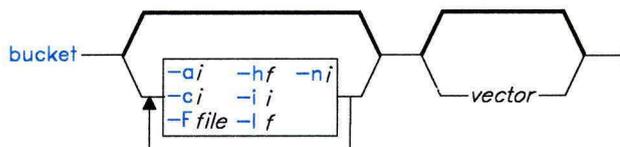
```
subset -FB,p1 A
```

For each element in B with a 1, output the corresponding element in A.

## Commands That Calculate Statistics (Summarizers)

### bucket

#### Syntax



OL805529

#### Description

The **bucket** command groups numbers into buckets. The input vector must be sorted. The output vector consists of odd (parenthesized) elements that are bucket limits and even elements that are bucket counts. The count is the number of elements greater than or equal to the lowest limit and less than or equal to the higher limit. Unless otherwise specified, bucket limits are generated based on the input data and the following rule:

$$\#buckets = 1 + \log_2(\#elements)$$

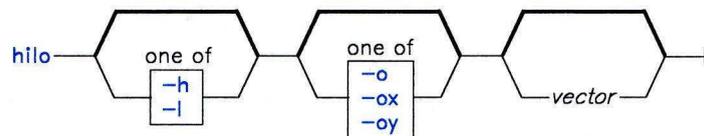


**Example**

To obtain correlation coefficients:

```
cor -Ffilea olddata newdata
```

This outputs the ordinary correlation coefficients between vectors `filea` and `olddata` and vectors `filea` and `newdata`.

**hilo**

OL805531

**Description**

The **hilo** command finds high and low values across all of the input vectors.

- h** Finds the high value only.
- l** Finds the low value only.
- o** Outputs the high and low values in option form (suitable for **plot**).
- ox** Outputs the high and low values in option form with **x** prefixed.
- oy** Outputs the high and low values in option form with **y** prefixed.

**Example**

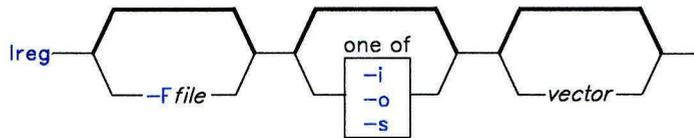
To find the lowest value:

```
hilo -ox,l file1 file2
```

This finds the lowest value in vectors `file1` and `file2` and outputs it with `xl` prefixed to it.

## lreg

### *Syntax*



OL805532

### *Description*

The `lreg` command provides linear regression. Output is the slope and intercept from a least squares linear regression of each vector on a base vector. The default base vector is the ascending positive integers from zero.

### *Flags*

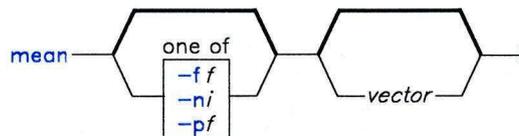
- `-Ffile` Specifies a file containing the first vector.
- `-i` Outputs only the intercept.
- `-o` Outputs the slope and intercepts in option form (suitable for **siline**).
- `-s` Outputs only the slope.

### *Example*

To output only the intercept:

```
lreg -Fbase,i mydata
```

This outputs the intercept from the linear regression of vector `mydata` on base vector `base`.

**mean****Syntax**

OL805533

**Description**

The **mean** command calculates the (trimmed) arithmetic mean.

**Flags**

**-ff** Specifies the fraction of elements to trim from each end. This is calculated as follows:

$$(1/f) k$$

where  $k$  is the total number of elements.

**-ni** Specifies the number of elements to trim from each end.

**-pf** Specifies the percentage of elements to trim from each end.

**Example**

To output the mean:

```
mean -p.25 mydata
```

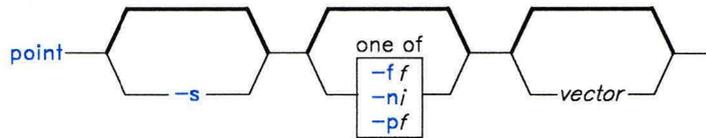
This outputs the mean of the middle 50% of the elements of `mydata`; that is, `mydata` is trimmed by 25% of its elements from both ends.

## stat

---

### point

#### *Syntax*



OL805534

#### *Description*

Output from the **point** command is a linearly interpolated value from the empirical cumulative density function for the input vector. By default, **point** returns the median (50% point).

#### *Flags*

- ff** Returns the  $(1/f)*100$  percent point.
- ni** Returns the  $i$ th element.
- pf** Returns the  $f*100$  percent point.
- s** Specifies that the input has been sorted.

#### *Example*

To output the 25% point:

```
point -s,p.25 mydata
```

### prod

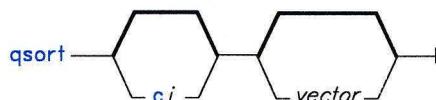
#### *Syntax*

```
prod — |
```

OL805556

#### *Description*

The **prod** commands calculates an internal product. Output is the product of the elements in the input vectors.

**qsort****Syntax**

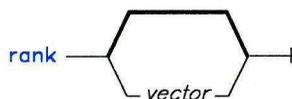
OL805536

**Description**

The `qsort` command does a quick sort. Output is a vector of the elements from the input vector in ascending order.

**Flag**

`-ci` Specifies the number of columns per line of output (5 by default).

**rank****Syntax**

OL805535

**Description**

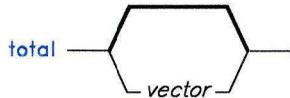
The `rank` command ranks vectors. Output is the number of elements in each input vector.

## stat

---

### total

#### *Syntax*



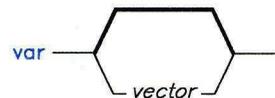
OL805537

#### *Description*

The **total** command calculates a sum total. Output is the sum total of the elements in the input vector(s).

### var

#### *Syntax*



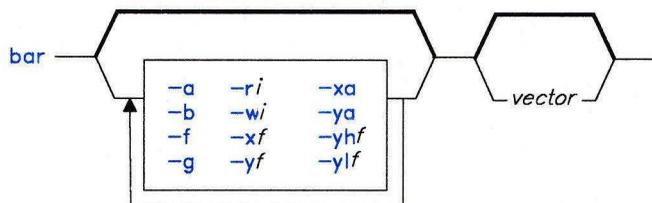
OL805538

#### *Description*

The **var** command calculates the variance.

## Commands That Produce Pictorial Output (Translators)

Input to these commands can be either a vector or a **GPS** object (a format for storing a picture). A picture is defined in a Cartesian plane of 64K points on each axis. The plane, or universe, is divided into 25 square regions numbered 1 to 25 from the lower left to the upper right.

**bar****Syntax**

OL805539

**Description**

The **bar** command builds a bar chart. It operates on an input vector, each element of which defines the height of a bar (y-axis). By default, the x-axis is labeled with positive integers, beginning at 1. For other labels, see **label**.

**Flags**

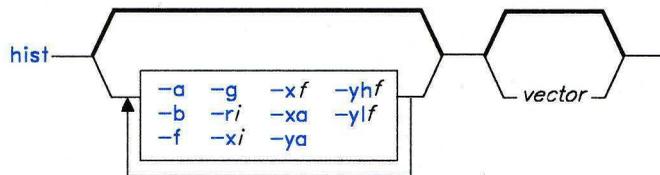
- a** Suppresses the axes.
- b** Plots the bar chart with bold weight lines (medium is the default weight).
- f** Does not build a frame around the plot area.
- g** Suppresses the background grid.
- ri** Puts the bar chart in **GPS** region *i*, where *i* is between 1 and 25 inclusive (13 by default).
- wi** Specifies the ratio of the bar width to center-to-center spacing expressed as a percentage (50 by default, giving equal bar width and bar space).
- xf**
- yf** Positions the bar chart in the **GPS** universe with the x-origin (y-origin) at *f*.
- xa**
- ya** Does not label the x-axis (y-axis).
- yhf** Specifies the y-axis high boundary.
- ylf** Specifies the y-axis low boundary.

**Example**

To produce a bar chart:

```
bar -r10,xa,w75 myfile
```

This outputs the bar chart described by vector `myfile`, located in region 10 of the **GPS** universe, with no x-axis labels. The bar width is 75% of center-to-center spacing.

**hist****Syntax**

OL805540

**Description**

The `hist` command builds a histogram. The input vector is the type produced by `bucket`, of odd rank, with odd elements being limits and even elements being bucket counts.

**Flags**

- `-a` Suppresses axes.
- `-b` Plots histogram with bold weight lines (the default weight is medium).
- `-f` Does not build a frame around the plot area.
- `-g` Suppresses the background grid.
- `-ri` Puts the histogram in **GPS** region *i*, where *i* is between 1 and 25 inclusive (13 by default).
- `-xf`
- `-yf` Positions the histogram in the **GPS** universe with the x-origin (y-origin) at *f*.
- `-xa`
- `-ya` Does not label the x-axis (y-axis).
- `-yhf` Specifies the y-axis high boundary.
- `-ylf` Specifies the y-axis low boundary.

### Example

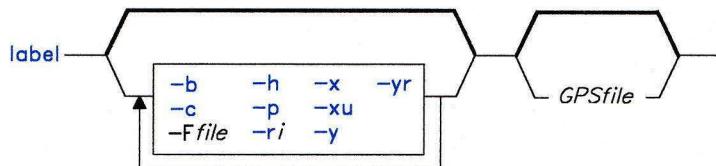
To produce a histogram:

```
hist -r5,ya myfile
```

This outputs the histogram described by vector `myfile` and locates it in region 5 of the GPS universe, with no y-axis labels.

## label

### Syntax



OL805541

### Description

The `label` command labels the axis of a **GPS** file.

### Flags

- b** Assumes that the input is a bar chart.
- c** Retains lowercase letters in labels; otherwise all letters are uppercase.
- Ffile** Specifies a label file. Each line of the file is taken as one label. Blank lines yield null labels. Either the **GPS** or the label file, but not both, can come from standard input.
- h** Assumes that the input is a histogram.
- p** Assumes that the input is an x-y plot. This is the default assumption.
- ri** Rotates labels *i* degrees. The pivot point is the first character.
- x** Labels the x-axis. This is the default action.
- xu** Labels the upper x-axis (the top of the plot).
- y** Labels the y-axis.
- yr** Labels the right y-axis (the right side of the plot).

**Examples**

1. To label a plot:

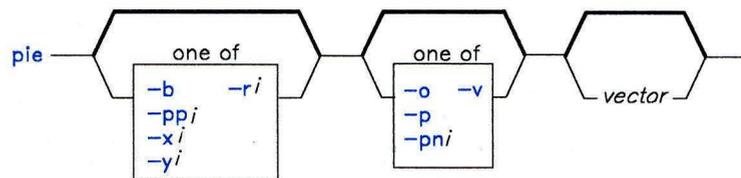
```
label -Flabs A.g
```

The file `A.g`, assumed to be an x-y plot, is labeled with labels from the file `labs`.

2. To label a plot from labels taken from standard input:

```
label -yr,r-45 A.g
```

The file `A.g` is labeled from the standard input. The labels are printed at 45 degrees below the horizontal.

**pie****Syntax**

OL805542

**Description**

The **pie** command builds a pie chart. The input vector has a restricted format. Each input line represents a slice of the pies and has the following form:

```
[<i e f ccolor>] value [label]
```

with brackets indicating optional fields. The control field options have the following effects:

- i** The slice will not be drawn, though a space will be left for it.
- e** The slice is “exploded” or moved away from the pie.
- f** The slice is filled. The angle of fill lines depends on the color of the slice.
- ccolor** The slice is drawn in the specified color rather than the default black. Legal values are **b** (black), **r** (red), **g** (green), and **u** (blue).

The pie is drawn with the value of each slice printed inside and the label printed outside.

---

### *Flags*

- b** Draws pie chart with bold weight lines (the default weight is medium).
- o** Places output values around the outside of the pie.
- p** Expresses output values as a percentage of the total pie.
- pni** Expresses output values as a percentage, but the total of the percentages equals *i* rather than 100.
- ppi** Draws only *i* percent of the pie.
- ri** Puts the pie chart in **GPS** region *i*, where *i* is between 1 and 25 inclusive (13 by default).
- v** Does not output values.
- xi**
- yi** Positions the pie chart in the **GPS** universe with x-origin (y-origin) at *i*.

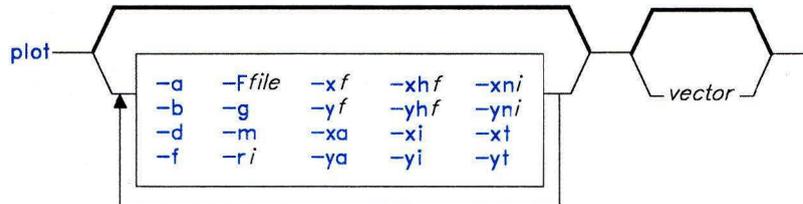
### *Example*

To draw a pie chart:

```
pie -pp80,pn80 chartfile
```

This draws the pie chart specified by `chartfile` in 80% of a circle and outputs the values as percentages of that total 80 percent.

## plot

*Syntax*

OL805543

*Description*

The **plot** command plots a graph. The input vectors contain the y values of an x-y graph. Values for the x-axis come from the file specified by **-F**. Axis scales are determined from the first vector plotted.

*Flags*

- a** Suppresses the axes.
- b** Plots the graph with bold weight lines (medium is the default weight).
- d** Does not connect plotted points (this implies **-m**).
- f** Does not build a frame around the plot area.
- Ffile** Uses the specified file for x values; otherwise the positive integers are used. You can specify this flag more than once, causing a different set of x values to be paired with each input vector. If there are more input vectors than sets of x values, the last set applies to the remaining vectors.
- g** Suppresses the background grid.
- m** Marks the plotted points.
- ri** Puts the graph in **GPS** region *i*, where *i* is between 1 and 25 inclusive (13 by default).
- xf**
- yf** Positions the graph in the **GPS** universe with the x-origin (y-origin) at *f*.
- xa**
- ya** Does not label the x-axis (y-axis).

- 
- xhf**
  - yhf** Specifies the x-axis (y-axis) high boundary.
  - xlf**
  - ylf** Specifies the x-axis (y-axis) low boundary.
  - xni**
  - yni** Specifies the approximate number of ticks on the x-axis (y-axis).
  - xt**
  - yt** Omits the x-axis (y-axis) title.

### ***Examples***

1. To plot against the positive integers:

```
plot plotdata
```

2. To customize x- and y-axes:

```
plot -r5,y10,xa,Fxfile yfile
```

This plots vector `yfile` against vector `xfile`, with y-axis ticks beginning at zero, no x-axis labels printed, and the plot placed in region 5 of the **GPS** universe.

```
plot -'hilo -oy filea fileb' filea fileb
```

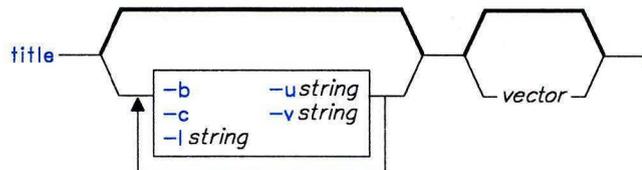
This plots vectors `filea` and `fileb` against the positive integers, with y-axis ticks going from the lowest to the highest values in the two vectors.

```
plot -Ffilea,Ffileb filec filed filee
```

This plots vectors `filec` against `filea`; `filed` and `filee` against `fileb`. The y-axis scale is determined from `filec`; the x-axis scale from `filea`.

title

*Syntax*



OL805544

*Description*

The `title` command prefixes a title to a vector or appends one to a **GPS** object.

*Flags*

- `-b` Makes the **GPS** title bold.
- `-c` Retains lowercase letters in the title; otherwise all letters are uppercase.
- `-l string` Uses the specified string as a **GPS** lower title.
- `-ustring` Uses the specified string as a **GPS** upper title.
- `-vstring` Labels a vector with the specified string.

**Related Information**

The following commands: “**ged**” on page 463, “**graphics**” on page 497, and “**spline**” on page 972.

The `gps` file in *AIX Operating System Technical Reference*.

---

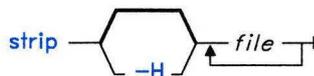
# strip

---

## Purpose

Removes symbol and line number information from a common object file.

## Syntax



OL805265

## Description

The **strip** command removes the symbol table and line number information from common object files, including archive libraries. Once you use this command, symbolic debugging of the file is difficult; therefore, you should normally run **strip** only on production modules that you have debugged and tested. Using **strip** reduces the file storage overhead required by an object file.

For each object module, **strip** removes all symbol table information. For each archive, **strip** removes the local symbol table information from each member.

You can restore a stripped symbol table to an archive or library file by using the **ar -s** command.

## Flag

**-H** Removes the object file header as well as all symbol table information.

## Files

/usr/tmp/strip\*

## Related Information

The following commands: “**ar**” on page 55, “**as**” on page 61, “**cc**” on page 140, “**dump**” on page 366, “**ld**” on page 557, “**nm**” on page 705, and “**size**” on page 949.

The **ar** and **a.out** files in *AIX Operating System Technical Reference*.

# stty

---

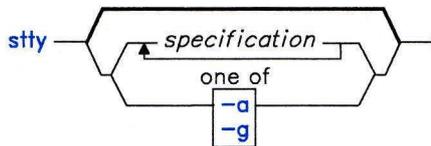
## stty

---

### Purpose

Sets, resets, or reports work station operating parameters.

### Syntax



OL805266

### Description

The `stty` command sets certain work station I/O options for the device that is the current standard input. If you run it without any *specifications*, `stty` writes to standard output information about any system adapters installed and reports the settings of certain options.

If you list any work station *specifications*, `stty` sets or resets the specified work station options.

You can find detailed information about the modes listed in the first six of the following groups in the discussion of the **termio** special facility in *AIX Operating System Technical Reference*. The last group contains options produced by combining options in the first six groups.

**Note:** The `stty` command does not make compatibility checks on any parameter combinations.

### Flags

- a Writes the current state of all option settings to standard output.
- g Writes option settings to standard output in a form usable by another `stty` command.

## Specifications

### Control Modes

The following options apply only when your work station connects to the system through an asynchronous line adapter. See **asy** in *AIX Operating System Technical Reference* for detailed information about this group.

**parenb (-parenb)** Enables (disables) parity generation and detection.

**parodd (-parodd)** Selects odd (even) parity.

**cs5 cs6 cs7 cs8** Selects character size.

**Note:** See **termio** in *AIX Operating System Technical Reference* for additional information on character size.

**0** Hangs up phone line immediately.

**50 75 110 134 150 300 600 1200 1800 2400 4800 9600 19200 19.2 38400 38.4 exta extb**  
Sets the work station speed to the specified number of bits per second (**exta**, 19200, and 19.2 are synonyms; **extb**, 38400, and 38.4 are synonyms). Regardless of the baud rate, the software requires that terminals generate and support the ASCII character set.

**hupcl (-hupcl)**

**hup (-hup)** Hangs up (does not hang up) dial-up connection on the last close.

**cstopb (-cstopb)** Selects 2 (1) stop bits per character.

The next two options apply to all work stations, regardless of the line adapter:

**cread (-cread)** Enables (disables) the receiver.

**local (-local)** Assumes a line without (with) modem control. The status of **local** is displayed when you use the **-a** flag. You cannot change this status with the **stty** command.

### Input Modes

**ignbrk (-ignbrk)** Ignores (does not ignore) **BREAK** on input.

**brkint (-brkint)** Signals (does not signal) **INTR** on break.

**ignpar (-ignpar)** Ignores (does not ignore) parity errors.

**parmrk (-parmrk)**  
Marks (does not mark) parity errors.

<b>inpck (-inpck)</b>	Enables (disables) input parity checking.
<b>istrip (-istrip)</b>	Strips (does not strip) input characters to 7 bits.
<b>inlcr (-inlcr)</b>	Maps (does not map) <b>NL</b> to <b>CR</b> on input.
<b>igncr (-igncr)</b>	Ignores (does not ignore) <b>CR</b> on input.
<b>icrnl (-icrnl)</b>	Maps (does not map) <b>CR</b> to <b>NL</b> on input.
<b>iucLC (-iucLC)</b>	Maps (does not map) uppercase alphabetic characters to lowercase.
<b>ixon (-ixon)</b>	Enables (disables) START/STOP output control. Once START/STOP output control has been enabled, you can pause output to the work station by pressing <b>Ctrl-S</b> and resume output by pressing <b>Ctrl-Q</b> .
<b>ixany (-ixany)</b>	Allows any character (only <b>Ctrl-Q</b> ) to restart output.
<b>ixoff (-ixoff)</b>	Sends (does not send) START/STOP characters when the input queue is nearly empty/full.

## Output Modes

<b>opost (-opost)</b>	Processes output (does not process output; that is, it ignores all other output options).
<b>olcuc (-olcuc)</b>	Maps (does not map) lowercase alphabetic characters to uppercase on output.
<b>onlcr (-onlcr)</b>	Maps (does not map) <b>NL</b> characters to <b>CR-NL</b> characters.
<b>ocrnl (-ocrnl)</b>	Maps (does not map) <b>CR-NL</b> characters to <b>NL</b> characters.
<b>onocr (-onocr)</b>	Does not (does) output <b>CR</b> characters at column zero.
<b>onlret (-onlret)</b>	On the terminal, <b>NL</b> performs (does not perform) the <b>CR</b> function.
<b>ofill (-ofill)</b>	Uses fill characters (uses timing) for delays.
<b>ofdel (-ofdel)</b>	Uses <b>DEL (NUL)</b> characters for fill characters.
<b>cr0 cr1 cr2 cr3</b>	Selects style of delay for <b>CR</b> characters.
<b>nl0 nl1</b>	Selects style of delay for <b>NL</b> characters.

**tab0 tab1 tab2 tab3**

Selects style of delay for horizontal tabs.

**bs0 bs1**

Selects style of delay for backspaces.

**ff0 ff1**

Selects style of delay for form feeds.

**vt0 vt1**

Selects style of delay for vertical tabs.

**Local Modes****isig (-isig)**

Enables (disables) the checking of characters against the special control characters **INTR** and **QUIT**.

**icanon (-icanon)**

Enables (disables) *canonical input* (canonical input allows input-line editing with the **ERASE** and **KILL** characters).

**xcase (-xcase)**

Echoes (does not echo) uppercase characters on input, and displays uppercase characters on output with a preceding \ (backslash).

**echo (-echo)**

Echoes (does not echo) every character typed.

**echoe (-echoe)**

Echoes (does not echo) the **ERASE** character as the backspace-space-backspace string.

**Note:** This mode does not keep track of column position, so you may get unexpected results when erasing tabs, escape sequences, and the like.

**echok (-echok)**

Echoes (does not echo) a **NL** character after a **KILL** character.

**lfkc (-lfkc)**

Functions the same as **echok**. This is an obsolete mode.

**echonl (-echonl)**

Echoes (does not echo) the **NL** character.

**noflsh (-noflsh)**

Does not clear (does clear) buffers after **INTR** or **QUIT**.

## Control Assignments

*control-character c* Set *control-character* to *c*, where *control-character* is **erase**, **kill**, **intr**, **quit**, **eof**, **eol**, **min**, or **time**. (Use **min** and **time** with **-icanon**.) If *c* is in the form `\^c` (backslash circumflex *c*), then its value is the corresponding **Ctrl** character. A `\^?` (backslash circumflex question mark) is interpreted as **DEL**. A `\^-` (backslash circumflex minus) is interpreted as undefined.

### **enhdedit (-enhdedit)**

Enters (leaves) the enhanced line editing discipline (see the **termio** special facility in *AIX Operating System Technical Reference*).

**Note:** When Japanese Language Support is installed, **enhdedit** is not supported.

### **ascedit (-ascedit)**

Enters (leaves) the ASCII keyboard mode for **dosedit**.

### **line i**

Sets the line discipline. *i* can be either 0 or 1. `stty line 0` is the same as `stty -enhdedit`. `stty line 1` is the same as `stty enhdedit`.

## Screen Length

### **page (-page)**

Pauses (does not pause) during output after each screen displayed. Typing any character during the pause causes output to resume. Typing a space during the pause causes output to continue uninterrupted until the next command is entered.

### **length n**

Sets screen length to *n* lines, where *n* is an integer from 1 through 255. An automatic pause in output occurs after *n* lines if **page** is enabled.

## Combination Modes

### **evenp | parity**

Enables **parenb** and **cs7**.

### **oddp**

Enables **parenb**, **cs7**, and **parodd**.

### **-parity, -evenp, -oddp**

Disables **parenb** and sets **cs8**.

### **raw (-raw | cooked)**

Enables (disables) **raw** input and output (no **ERASE**, **KILL**, **INTR**, **QUIT**, **EOT**, or output processing).

---

<b>nl (-nl)</b>	Unsets (sets) <b>icrnl</b> and <b>onlcr</b> . Specifying <b>-nl</b> sets <b>icrnl</b> and <b>onlcr</b> and also unsets <b>inlcr</b> , <b>igncr</b> , <b>ocrnl</b> , and <b>onlret</b> .
<b>lcase (-lcase)</b> <b>LCASE (-LCASE)</b>	Sets <b>xcase</b> , <b>iucrc</b> , and <b>olcuc</b> . (Used for work stations with uppercase characters only.)
<b>tabs (-tabs   tab3)</b>	Preserve tabs (expand to spaces) when printing.
<b>ek</b>	Sets <b>ERASE</b> and <b>KILL</b> characters to <b>Ctrl-H</b> and <b>Ctrl-U</b> , respectively.
<b>sane</b>	Resets parameters to "reasonable" values.
<b>term</b>	Sets all parameters according to work station type <i>term</i> , where <i>term</i> is one of <b>tty33</b> , <b>tty37</b> , <b>vt05</b> , <b>tn300</b> , <b>ti700</b> , or <b>tek</b> .

### Terminal Mapping

<b>dmap</b> <i>mapname</i>	Deactivates a loaded map. Any user can deactivate a map.
<b>imap</b> <i>mapname</i>	Loads and activates <code>/etc/nls/termmap/<i>mapname</i>.in</code> as the terminal input map. If no <i>mapname</i> is defined, <b>imap</b> activates the previously specified map.  <b>Note:</b> You must be a superuser or a member of the system group to load a map. Any user can activate a loaded map with <b>imap</b> .
<b>omap</b> <i>mapname</i>	Loads and activates <code>/etc/nls/termmap/<i>mapname</i>.out</code> as the terminal output map. If no <i>mapname</i> is defined, <b>omap</b> activates the previously specified map.  <b>Note:</b> You must be a superuser or a member of the system group to load a map. Any user can activate a loaded map with <b>omap</b> .

### Examples

1. To display a short listing of your work station configuration:

```
stty
```

This lists settings that differ from the defaults.

2. To display a full listing of your work station configuration:

```
stty -a
```

3. To enable a key sequence that stops listings from scrolling off the screen:

```
stty ixon ixany
```

This sets **ixon** mode, which lets you stop runaway listings by pressing **Ctrl-S**. The **ixany** parameter allows you to resume the listing by pressing any key. The normal

work station configuration includes `ixon` and `-ixany`, which allows you to stop a listing with **Ctrl-S**, but only **Ctrl-Q** will restart it.

4. To prevent all listings from scrolling off the screen:

```
stty page length 24
```

This sets page mode with a page (screen) length of 24 lines. When a listing is more than 24 lines long, the system pauses after each page. It beeps, reminding you to press any key (except the **Spacebar**) to view the next page. Press the **Spacebar** to let the rest of the listing scroll off the screen and get to the end. Paging then resumes with the next listing.

5. To reset the configuration after it has been messed up:

```
Ctrl-J stty sane echo -tabs Ctrl-J
```

Sometimes the information displayed on the screen may look strange, or the system won't respond when you press the **Enter** key. This can happen when you use `stty` with parameters that are incompatible or that do things you don't understand. It can also happen when a screen-oriented text editor ends abnormally and doesn't have a chance to reset the work station configuration.

Entering `stty sane` sets a reasonable configuration, but it may differ slightly from your normal configuration. That is why this example also includes two commonly used parameters, `echo` (erase characters as you backspace over them) and `-tabs` (expand tab characters to spaces on the display screen).

Press **Ctrl-J** before and after the command instead of **Enter**. The system usually recognizes **Ctrl-J** when the parameters that control the **Enter** key processing are messed up.

6. To save and restore the work station's configuration:

```
OLDCONFIG=`stty -g`          # save configuration
stty -echo                  # do not display password
echo "Enter password: \c"
read PASSWD                 # get the password
stty $OLDCONFIG             # restore configuration
```

This saves the work station's configuration, turns off echoing, reads a password, and restores the original configuration. The `` . . . `` (grave accents) in the first command tell the shell to insert the standard output of `stty -g` into the `OLDCONFIG= . . .` command. This is called *command substitution*. For more information, see "Command Substitution" on page 925.

The `stty -echo` turns off echoing, which means that the password does not appear on the screen when you type it at the keyboard. This has nothing to do with the `echo` command, which displays a message on the screen.

## **Related Information**

The following command: “**tabs**” on page 1041.

The **ioctl** system call and the **terminfo** and **config** files in *AIX Operating System Technical Reference*.

The discussion of **stty** and “Overview of International Character Support” in *IBM RT Managing the AIX Operating System*.

## su

---

## su

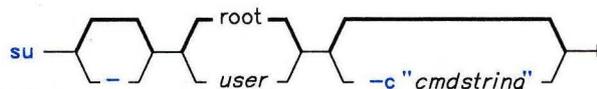
---

### Purpose

Obtains the privileges of another user, including superuser authority.

### Syntax

```
su -c "cmdstring" user
```



OL805267

### Description

The **su** command runs a shell and lets you operate there with the privileges of the specified *user* (by default *root*). If you are not already operating with superuser authority, **su** prompts for the password associated with *user* before granting you these privileges. Upon successful authentication, **su** will set the audit classes for the new shell to those specified in */etc/security/passwd*.

If the terminal is in a trusted state, the shell will be **tsh** and the prompt will be set to **tsh**.

If you use **su** to become the superuser (*user* is *root*), **su** sets the **PATH** variable to */bin:/etc:/usr/bin* and changes the prompt to **#** (pound sign). Notice that this **PATH** does not include the current directory.

To restore your normal privileges, press **END OF FILE (Ctrl-D)**. This action ends the shell called by **su** and returns you to the previous shell and ID.

If you need to run only one command as *user*, you can run the desired command by including it (along with any of its associated flags) on the command line as an argument to the shell **-c** flag (see “**sh**” on page 913 for a description of this flag). In this case, **su** calls **sh** to run the command and then exits automatically.

Each time someone uses **su** to become the superuser, **su** writes a record in the file */usr/adm/sulog*, creating this file if necessary.

**Note:** If the **-c** flag is not specified, **su** execs the shell listed in the shell field of the */etc/passwd* file. If the **-c** flag is specified, **su** ignores the **passwd** file entry and runs */bin/sh*. All exported environment variables are available unless you use the **-** flag when you call **su**.

---

## Flags

Creates the user's login environment by calling the new shell as a **login shell** (see "sh" on page 913). It reads the system **profile** file and the user's **\$HOME/.profile** file. The environment variables **NLLDATE** and **NLTIME** control the appearance of the date and time. The **TERM** and **TZ** variables are an exception. They are preserved at their current values. These variables are normally set by **init** or **getty** prior to login; as a result, **su** handles them differently.

**Note:** This flag modifies the environment of the current shell only if the optional program named in the shell field of the **passwd** file is a program like **sh**.

- c "cmdstring"** Runs the **/bin/sh** shell, processes the specified *command*, and then exits the shell. This flag causes **su** to ignore the shell specified in the **passwd** file.

## Examples

1. To obtain superuser authority:

```
su
```

This runs a subshell with the effective user ID and privileges of user **root**. The **su** command asks for a password, as if you were logging in as **root**. Now the commands you run have superuser authority. Press **END OF FILE (Ctrl-D)** to end the subshell and return to your original shell session and privileges.

2. To obtain jim's privileges:

```
su jim
```

This runs a subshell with the effective user ID and privileges of **jim**.

3. To set up the environment as if you had logged in as jim:

```
su - jim
```

This runs a subshell with the effective user ID and privileges of **jim**. The **-** (minus) causes the shell variable **LOGNAME** to be set to **jim**, **HOME** to be set to the path name of **jim**'s home directory, and **jim**'s **\$HOME/.profile** shell procedure file to be run before prompting for the first shell command.

4. To run a single command with superuser authority:

```
su root -c "backup -9 -u"
```

This runs the shell command `backup -9 -u` with superuser authority (if you know the password assigned to `root`).

### Related Information

The following command: “**sh**” on page 913.

---

## sum

---

### Purpose

Displays the checksum and block count of a file.

### Syntax

```
sum -r file
```

OL805268

### Description

The **sum** command reads *file* and calculates a 16-bit checksum for the *file* and the number of blocks in the file. The checksum and number of blocks are written to standard output. The **sum** command is generally used to determine if a file that has been copied or communicated over transmission lines is an exact copy of the original.

### Flag

- r** Uses an alternate algorithm to compute the checksum (rigorous byte-by-byte computation rather than the default word by word computation).

### Example

To display the checksum of, and the number of blocks in `datafile`:

```
sum datafile
```

If the checksum of `datafile` is 1605 and if the file contains 3 blocks, then **sum** displays:

```
1605 3 datafile
```

### Related Information

The following command: “**wc**” on page 1211.

## sync

---

## sync

---

### Purpose

Updates the superblock and writes buffered files to the fixed disk.

### Syntax

`sync` —

OL805221

### Description

The **sync** command runs the **sync** system primitive. If you have to stop the system, you must run **sync** to ensure file system integrity. **sync** writes all unwritten system buffers to disk. This includes modified superblocks, modified i-nodes, delayed block I/O, and read-write mapped files.

**Note:** The writing, although scheduled, is not necessarily complete upon return from the **sync** system call.

### Related Information

The **sync** system call in *AIX Operating System Technical Reference*.

---

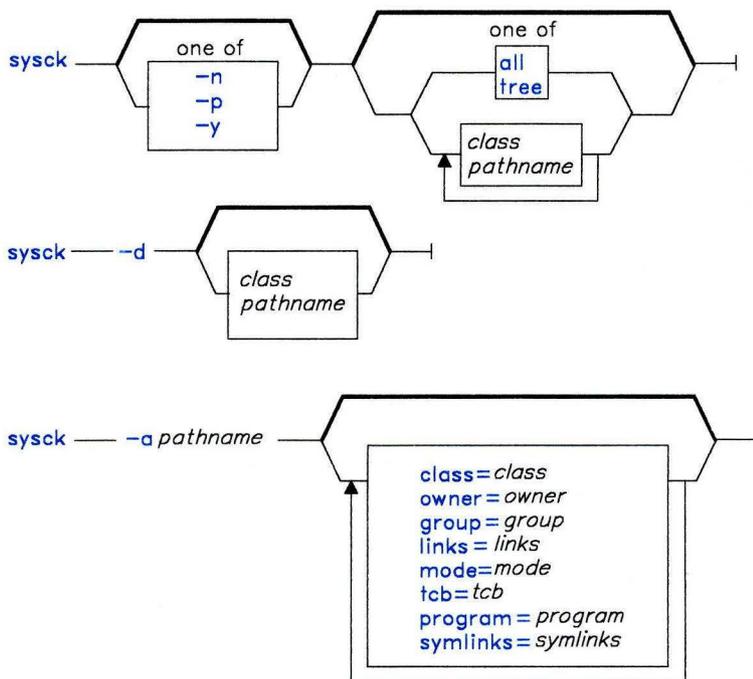
# sysck

---

## Purpose

Verifies the secure system state

## Syntax



OL805503

## Description

The `sysck` command checks the installation of files relevant to security. Each such file must have a corresponding stanza in `/etc/security/sysck.cfg`. The `sysck` command also adds or deletes file descriptions from `/etc/security/sysck.cfg`.

When invoked with no flags and with no arguments, the `sysck` command prints a synopsis of its usage.

## Adding a File Description

When invoked with the **-a** flag, the **sysck** command adds or modifies a stanza in the **/etc/security/sysck.cfg** file. The added or modified stanza describes the file specified by *pathname*. The file attributes are taken from:

- The old stanza for the file (if it exists).
- The file (which must exist).
- The *attribute = value* arguments on the command line (if there are any).

You may specify that an attribute is not to appear in the stanza by specifying an empty string as the value for the attribute. See the discussion of the **sysck.cfg** file in *AIX Operating System Technical Reference* for a description of the attributes recorded in **/etc/security/sysck.cfg**.

## Deleting a File Description

When invoked with the **-d** flag, the **sysck** command deletes the specified stanzas from **/etc/security/sysck.cfg**. A stanza may be deleted by specifying its *pathname* or by specifying the value of its **class** attribute.

## Checking a File Description

When invoked with neither the **-a** nor **-d** flags, the **sysck** command checks the installation of files described in **/etc/security/sysck.cfg**. Any inconsistency can be fixed by changing the file to match the description in **/etc/security/sysck.cfg**.

The arguments to the **sysck** command specify which stanzas are to be checked:

- pathname* Specifies the name of a stanza. This must be an absolute path name (i.e., begin with a /).
- class* Specifies all stanzas whose **class** attribute has the value *class*. Note that a class name cannot begin with a /.
- all** Specifies all stanzas.
- tree** Specifies all stanzas. When **tree** is specified, the **sysck** command not only checks the installation of files described in **/etc/security/sysck.cfg**, but it also checks that there are no other files in the file system which should be described by **/etc/security/sysck.cfg**.

The following checks are performed on each file *not* described by a stanza in **/etc/security/sysck.cfg**:

- If the file is a hard link to a file described in **/etc/security/sysck.cfg**, but is not listed in the value of the **links** attribute, the link is removed.
- If the file is a symbolic link to a file described in **/etc/security/sysck.cfg**, but is not listed in the value of the **symlinks** attribute, the symbolic link is removed.

- If the file is a regular file, the file owner ID is 0, and the file mode has the **S\_ISUID** bit set, the **S\_ISUID** bit is cleared.
- If the file is a regular file, the file group ID is 0, and the file mode has the **S\_ISGID** and **S\_IXGRP** bits set, the **S\_ISGID** bit is cleared.
- If the file has the **tcb** attribute set, the **tcb** attribute is cleared.
- If the file is a block or character device, all permission bits of the file are set to 0.

The action to be taken when a configuration error is found is determined by any **-n**, **-p**, or **-y** flag supplied to the **sysck** command.

The following checks are performed if the stanza has the specified attribute. If an inconsistency is found, **sysck** performs the action specified by the flag supplied with the **sysck** command.

<b>class</b>	The value of this attribute is the name of a group of stanzas. The <b>sysck</b> command performs no checks based on the value of this attribute. The <b>class</b> attribute is used to specify a group of stanzas to be checked by the <b>sysck</b> command.
<b>owner</b>	The value of this attribute is a decimal user ID, or the login name of a user in <b>/etc/passwd</b> . The file owner ID should be this user ID.
<b>group</b>	The value of this attribute is a decimal group ID, or the name of a group in <b>/etc/group</b> . The file group ID should be this group ID.
<b>mode</b>	The value of this attribute can be an octal numeral, or a string in the form <b>-r-Sr-x--T</b> . The file mode should be this value.
<b>links</b>	The value of this attribute is a comma-separated list of additional links to the file.
<b>symlinks</b>	The value of this attribute is a comma-separated list of symbolic links to the file. Note that <b>sysck</b> will find additional symbolic links only if given the <b>tree</b> argument.
<b>tcb</b>	The value of this attribute is <b>true</b> or <b>false</b> . If <b>true</b> , the file should be flagged as part of the trusted computing base.
<b>program</b>	The value of this attribute is the full path name of a program that performs additional consistency checks on the file. Arguments to the program also appear with this attribute. Note that the value of an attribute must be enclosed in double quotes if it has embedded spaces. The <b>sysck</b> command will execute this program with any arguments preceded by the same flags ( <b>-n</b> , <b>-p</b> , or <b>-y</b> ) as were provided on the <b>sysck</b> command line.

## Consistency Checks

Two consistency check programs are supplied: **pwdchk** and **grpchk**.

### **/etc/security/pwdchk**

The **pwdchk** program checks the **/etc/passwd** and **/etc/security/passwd** files for internal and mutual consistency. Consistency checks are not applied to lines in **/etc/passwd** with start with - (minus) or + (plus).

The internal consistency checks for **/etc/passwd** include the following:

- Each line must have seven colon-separated fields. Any malformed entry is reported but not corrected.
- The *password* field of each entry must be ! (exclamation point). An invalid entry is reported and set to ! (exclamation point); if the *password* file is 13 or more characters, the leading 13 characters become the value of the **password** attribute in **/etc/security/passwd**.
- The *uid* field of each entry must be decimal numeral. An invalid entry is reported and the value of the **restrictions** attribute in **/etc/security/passwd** is set to **none**.
- The *gid* field of each entry must be decimal numeral which corresponds to a group described in **/etc/group**. An invalid entry is reported and the value of the **restrictions** attribute in **/etc/security/passwd** is set to **none**.
- The *directory* field must be **NULL** or a full path name. An invalid entry is reported but not corrected.
- The *shell* fields must be **NULL** or a full path name. An invalid entry is reported but not corrected.

The mutual consistency checks for **/etc/passwd** and **/etc/security/passwd** include the following:

- For every entry in **/etc/passwd**, there must be a stanza with the same user name in **/etc/security/passwd**. A stanza is created in **/etc/security/passwd** for any extra entries in **/etc/passwd**.
- For every entry in **/etc/security/passwd**, there must be a corresponding entry in **/etc/passwd**. Extra entries in **/etc/security/passwd** are reported and deleted.
- Each entry in **/etc/passwd** must have a unique user ID, unless **/etc/security/passwd** has a stanza for this user in which the **restrictions** attribute is **none**. Only one login account is allowed for each unique user ID unless specifically allowed by **restrictions = none** in **etc/security/passwd**. Subsequent entries with the same user ID as a previous entry are reported and the value of the **restrictions** attribute is set to **nologin**.

## **/etc/security/grpchk**

The **grpchk** program checks the **/etc/group** and **/etc/security/group** files for internal and mutual consistency.

The internal consistency checks for **/etc/group** include the following:

- Each line must have three colon-separated fields. Any malformed entry is reported, but not corrected.
- Each group name must be unique. A duplicate entry is reported, but not corrected.
- The *gid* field of each entry must be unique. An invalid entry is reported, but not corrected.
- Each user listed as a member of the group must have an entry in **/etc/passwd**. An unknown user name is reported, but not corrected.

The mutual consistency checks for **/etc/group** and **/etc/security/group** include the following:

- For every entry in **/etc/group**, there must be a stanza with the same group name in **/etc/security/group**. A stanza is created in **/etc/security/group** for any extra entries in **/etc/group**.
- For every entry in **/etc/security/group**, there must be a stanza with the same group name in **/etc/group**. A stanza is created in **/etc/group** for any extra entries in **/etc/security/group**.

---

## **Japanese Language Support Information**

If Japanese Language Support is installed on your system, this command is not available.

---

## **Flags**

- a Adds (to **/etc/security/sysck.cfg**) the description of a file.
- d Deletes (from **/etc/security/sysck.cfg**) the description of the files specified by the arguments.
- n Checks the installation of the files specified by the arguments. Any inconsistencies are reported. No changes will be made to either the file or to **/etc/security/sysck.cfg**.
- p Checks the installation of the files specified by the arguments. Any inconsistencies are **not** reported. Changes are made to files so they match the description in **/etc/security/sysck.cfg**.
- y Checks the installation of the files specified by the arguments. Any inconsistencies are reported. Changes are made to files so they match the description in **/etc/security/sysck.cfg**.

## sysck

---

### Example

The following command assures all files described by `/etc/security/sysck.cfg` are properly installed:

```
sysck -y all
```

The following command can be used to construct an `/etc/security/sysck.cfg` file which reflects the current state of the system:

```
sysck tree
```

### Files

<code>/etc/security/sysck.cfg</code>	Defines installation assertions for security relevant files and directories.
--------------------------------------	--

### Related Information

The following file format: **sysck.cfg** in *AIX Operating System Technical Reference*.

The discussion of the trusted computing base in *Managing the AIX Operating System*.

---

# syslogd

---

## Purpose

Reads and logs messages.

## Syntax

```

/etc/syslogd -fconfigfile -mmarkinterval -d

```

AJ2FL148

## Description

The **syslogd** command reads and logs messages into a set of files described by the configuration file **/etc/syslog.config**. This daemon configures itself when it starts up and whenever it receives a hangup signal.

Each message read by **syslogd** is one line. A message can contain a priority code (marked by a number in < > brackets at the beginning of the line) and message text. Priorities are defined in **sys/syslog.h**. The **syslogd** command reads from the AIX domain socket **/dev/log** or from an Internet domain socket specified in **/etc/services**.

Each line in the **syslogd** configuration file must consist of two parts:

- A selector to determine the message priorities to which the line applies
- An action.

The two fields must be separated by one or more tabs. Here is an example of the line in a configuration file:

```
mail.info;*.notice           /usr/spool/adm/syslog
```

The first part, the selector, is semicolon-separated list of priority specifiers. Each priority specifier consists of a facility describing the part of the system that generated the message, a . (period), and a level indicating the severity of the message. Symbolic names may be used and an \* (asterisk) specifies all facilities. All messages of the specified level or higher (greater severity) are selected. In the previous example, **syslogd** selects the mail facility at the info level (or higher) and all facilities at the notice level (or higher).

## syslogd

---

More than one facility may be selected using commas to separate them. For example:

```
*.emerg;mail,daemon.crit
```

selects all facilities at the `emerg` level (or higher) and the `mail` and `daemon` facilities at the `crit` level (or higher).

Known facilities and levels recognized by **syslogd** are those listed under **syslog** in the *AIX Operating System Technical Reference*. When you specify the name of a facility or level in a **syslogd** configuration file, omit the **LOG\_** prefix used by **syslog** in the name. For example, **syslog** lists **LOG\_DEBUG** as the lowest level. To specify this level in a **syslogd** configuration file, specify `debug`.

In addition to these facilities, there is a **mark** facility. This facility has messages at priority **info** sent to it every 20 minutes. You can change the mark time interval with the **-m** flag. The **mark** facility is not enabled by a facility field containing an asterisk; you must explicitly enable it. For example:

```
kern,mark.debug
```

logs kernel messages and 20-minute marks of `debug` level (or higher).

The level `none` may be used to disable a particular facility. For example:

```
*.debug;mail.none
```

logs all messages except mail messages.

The second part of each line, the action, describes where the message is to be logged if the line is selected. There are four forms:

- A file name beginning with a leading `/` (Selected messages are appended to this file)
- A host name preceded by a `@` (Selected messages are forwarded to **syslogd** on the named host)
- A comma-separated list of users (Selected messages are written to those users, if they are logged in)
- An `*` (Selected messages are written to all logged-in users).

For example:

```
*.crit          /usr/adm/critical
kern.err        @nick
*.alert         bobbi,kristi
*.emerg        *
```

logs critical (or higher) messages into `/usr/adm/critical`, forwards kernel messages of error severity (or higher) to **syslogd** on the host `nick`, informs the users `bobbi` and `kristi` of any alert (or higher) messages, and informs all logged-in users of any emergency messages.

Blank lines and lines beginning with `#` are ignored.

The **syslogd** command creates the file `/etc/syslog.pid`, containing a single line with its process ID. This file can be used to kill or reconfigure **syslogd**. To bring **syslogd** down, it should be sent a terminate signal. For example:

```
kill 'cat /etc/syslog.pid'
```

## Flags

- d** Turns on debugging.
- f *configfile*** Specifies an alternate configuration file.
- m *markinterval*** Specifies the number of minutes between mark messages.

## Example

To start **syslogd** daemon and change the mark interval:

```
syslogd -m30
```

This command changes the mark interval to 30 minutes. If the configuration file contains:

```
kern,mark.notice      /usr/adm/notice
kern.err              @scott
*.info;mail.none     /usr/spool/adm/syslog
*.alert;auth.warning darlene
```

**syslogd** logs kernel messages and 30-minute marks at notice level (or higher) in the file `/usr/adm/notice`, forwards kernel messages at err level (or higher) to **syslogd** on the host `scott`, logs messages at info level (or higher) except mail messages in the file `/usr/spool/adm/syslog`, and informs the user `darlene` of any warning message (or higher) from the authorization system.

## Files

- `/etc/services` Contains definition of the Internet domain socket.
- `/etc/syslog.conf` Contains the configuration file.
- `/etc/syslog.pid` Contains the process id.
- `/dev/log` Contains AIX domain datagram log socket.

## Related Information

The **syslog** system call in *AIX Operating System Technical Reference*.

# tab

---

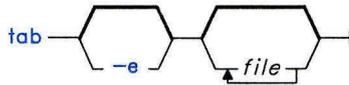
## tab, untab

---

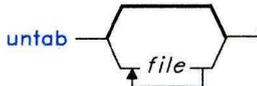
### Purpose

Changes spaces into tabs or tabs into spaces.

### Syntax



OL805069



OL805065

### Description

The **tab** command reads *files* (standard input by default), and replaces spaces in the input with tab characters wherever it can eliminate one or more spaces. It writes the resulting file back to *file* or, if the input was standard input, to standard output. **tab** assumes that the tab stops are set every eight columns starting with column nine.

The **untab** command reads *files* or standard input, replaces tabs in the input with space characters and writes back to the original file or to standard output.

### Flag

- e** Replaces only those spaces at the beginning of a line up to the first non-space character.

### Related Information

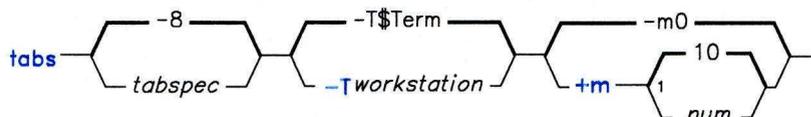
The following command: “**newform**” on page 686.

# tabs

## Purpose

Sets tab stops on work stations.

## Syntax



<sup>1</sup> Do not put a space between these items.

OL805381

## Description

The **tabs** command clears up to 20 previous tabs and sets up to 40 tabs on the work station according to the supplied *tabspec*. *tabspec* can be either a flag indicating an available code or column numbers. The available codes cover formats required by most structured programming languages.

When you use the **tabs** command, always see the leftmost column number as 1, even if your work station refers to it as zero (0).

If you do not specify a *tabspec*, the default value is -8.

## Tabspecs

- a Sets the tabs to 1, 10, 16, 36, and 72 (IBM S/370 Assembler first format)
- a2 Sets the tabs to 1, 10, 16, 40, and 72 (IBM S/370 Assembler second format)
- c Sets the tabs to 1, 8, 12, 16, 20, and 55 (COBOL normal format)
- c2 Sets the tabs to 1, 6, 10, 14, and 49 (COBOL compact format, columns 1-6 omitted).  
With this code, the first column position corresponds to card column 7. One space gets you to column 8, and a tab reaches column 12. Files using this code should include a format specification of:

```
<:t-c2 m6 s66 d:>
```

## tabs

---

For an explanation of format specifications, see the **fspec** file in *AIX Operating System Technical Reference*.

- c3** Sets the tabs to 1, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 50, 54, 58, 62, and 67 (COBOL compact format with more tabs than **-c2**. This is the recommended format for COBOL. Files using this code should include a format specification of:

```
<:t-c3 m6 s66 d:>
```

- f** Sets the tabs to 1, 7, 11, 15, 19, and 23 (FORTRAN).
- p** Sets the tabs to 1, 5, 9, 13, 17, 21, 25, 29, 33, 37, 41, 45, 49, 53, 57, and 61 (PL/I).
- s** Sets the tabs to 1, 10, and 55 (SNOBOL).
- u** Sets the tabs to 1, 12, 20, and 44.

In addition to the preset formats, three other types of *tabspecs* are available:

- num** Sets regularly repeating tabs at every *num*th column. (-8 is the standard AIX tab setting and the one required for use with the **nroff -h** flag.) Another special case is **-0**, which implies no tabs at all.
- num[,num] . . .** Sets tabs at the named column numbers (a comma-separated list in ascending order). You may specify up to 40 numbers. If any number except the first has a plus sign prefix, the prefixed number is added to the previous number for the next setting. Thus, the tab lists 1,10,20,30 and 1,10,+10,+10 provide the same tab settings.
- filep** Reads the first line of the named *filep* for a format specification. If it finds one, it sets tabs the same way. If it does not find a format specification, it sets tabs to the system default (**-8**). Use this *tabspec* to make sure that a file has the same tab settings as those in a file already correctly formatted.

## Flags

**Note:** If the same flag occurs more than once, only the last one takes effect.

- Tworkstation** Identifies the work station so that **tabs** can set tabs and margins correctly. *workstation* is one of the work stations listed under the **greek** command. If you do not provide a **-T** flag, **tabs** uses the shell variable **\$TERM**. If no *workstation* can be found, **tabs** tries a general value that works for most work stations.
- +mnum** Moves all tabs to the right *num* columns, and makes column *num*1 the left margin. If **m** is given without a value, 10 is assumed. The leftmost margin on most work stations is defined by **m0**.

## Related Information

The following commands: “**greek**” on page 499, “**nroff, troff**” on page 709, and “**troff**” on page 710.

The discussion of **term** and **environ** in *AIX Operating System Technical Reference*.

# tail

---

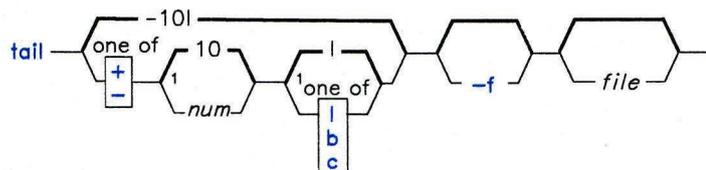
## tail

---

### Purpose

Writes a file to standard output, beginning at a specified point.

### Syntax



<sup>1</sup> Do not put a blank between these items.

OL805303

OL805308

### Description

The **tail** command writes the named *file* (standard input by default) to standard output, beginning at a point you specify. It begins reading at `+ [num]` lines from the beginning of *file* or `-[num]` lines from the end of *file*. The default *num* is 10. *num* is counted in units of lines, blocks, or characters, according to the subflag appearing after *num* (see the following flags).

### Flags

**-f** Does not end after it copies the line of the input file if the input file is not read from a pipe, but enters an endless loop in which it sleeps for a second and then attempts to read and copy further records from the input file. Thus, it can be used to monitor the growth of a file being written by another process.

`+ [num]l`  
`+ [num]b`  
`+ [num]c`

Begins reading *num* lines (**l**), blocks (**b**), or bytes (**c**) from the beginning of the input.

---

`-[num]l`  
`-[num]b`  
`-[num]c` Begins reading *num* lines (**l**), blocks (**b**), or bytes (**c**) from the end of the input.

---

### Japanese Language Support Information

`+ [num]k` Begins reading *num* characters (**k**) from the beginning of the input.  
`- [num]k` Begins reading *num* characters (**c**) from the end of the input.

The **c** flag in Japanese Language Support begins reading as closely as possible to the number of bytes requested, without breaking a 2-byte character. The number of characters in input containing SJIS characters may not equal the number of bytes. To get, or skip, precisely *num* characters, use the **k** flag.

---

## Examples

1. To display the last 10 lines of a file:

```
tail notes
```

2. To specify how far from the end to start:

```
tail -20 notes
```

This displays the last 20 lines of notes.

3. To specify how far from the beginning to start:

```
tail +200c notes | pg
```

This displays notes a page at a time starting with the 200th character from the beginning.

4. To follow the growth of a file:

```
tail -1 -f accounts
```

This displays the last line of accounts. Once a second, **tail** displays any lines that have been added to the file. This continues until stopped by pressing **INTERRUPT (Alt-Pause)**.

### Related Information

The following commands: “**dd**” on page 301 and “**pg**” on page 744.

The discussion of Japanese Language Support in *Japanese Language Support User's Guide*.

---

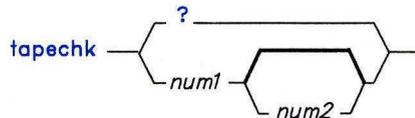
# tapechk

---

## Purpose

Performs consistency checking of the streaming tape device.

## Syntax



OL805445

## Description

The **tapechk** command performs rudimentary consistency checking on an attached streaming tape device. Some hardware malfunctions with a streaming tape drive can be detected by simply reading a tape. **tapechk** provides a way to perform tape reads on the file level.

Since the streaming tape drive cannot backspace over physical data blocks or files, **tapechk** rewinds the tape to its starting position prior to each check. You can specify numeric arguments to control the number of files checked or skipped. If you do not specify any arguments, **tapechk** rewinds the tape and checks only the first physical block.

**Note:** The **backup** command allows you to archive files selectively or as an entire file system. It writes data as a continuous stream terminated by a file mark, regardless of the number of files specified. The **tapechk** command perceives each stream of data as a single file. This is important when you specify numeric arguments with the **tapechk** command.

Although you can use **tapechk** on any streaming tape cartridge, it is primarily designed for checking tapes written by the **backup** command.

## Flags

*num1* Checks data for the next *num1* files.

*num2* Skips the next *num2* files from the beginning of the tape.

? Explains the format of the **tapechk** command.

**Note:** If you specify this argument, it must be the first argument.

# tar

---

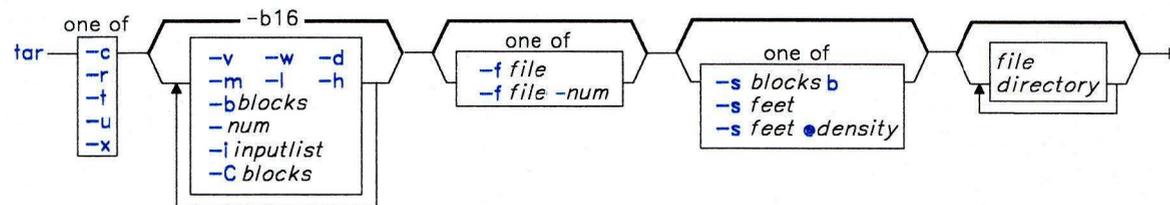
# tar

---

## Purpose

Manipulates archives.

## Syntax



OL805423

## Description

The **tar** command writes files to or retrieves files from an archival storage medium. The **tar** command looks for archives on the default device (usually tape), unless you specify another device with the **-f** flag. File names must not be longer than 100 bytes and must not contain blanks. Characters following the first blank are ignored.

When writing to an archive, **tar** uses a temporary file (**/tmp/tar\***) and maintains in memory a table of files with several links. You will receive an error message if **tar** cannot create the temporary file, or if there is not enough memory available to hold the link tables.

### Notes:

1. When the storage device is an ordinary file or a block special file, **-u** and **-r** flags backspace. However, raw magnetic tape devices do not support backspacing. So when the storage device is a raw magnetic tape, the **-u** and **-r** flags rewind the tape, open it, and then read it again.
2. Records are one block long on block magnetic tape, but they are typically less than half as dense on raw magnetic tape. As a result, although a blocked raw tape must be read twice, the total amount of tape motion is less than it is when reading one-block records from a block magnetic tape once.

3. The structure of a streaming tape device does not support the addition of information at the end of a tape. Consequently when the storage device is a streaming tape, the **-u** and **-r** flags are not valid options. An attempt to use these flags results in the error message `tar: Update and Replace options not valid for a streaming tape drive`.
4. There is no way to ask for any occurrence of a file other than the last.
5. There is no recovery from tape errors.

## Flags

You must supply one of the following five function flags to control the actions of **tar**:

- |          |   |
|----------|---|
| <b>c</b> | Creates a new archive and writes the <i>file</i> at the beginning of the archive.   |
| <b>r</b> | Writes the <i>file</i> at the end of the archive. Since the structure of a streaming tape device does not support the addition of information at the end of a tape, this option is not a valid flag when the archival storage device is a streaming tape.   |
| <b>t</b> | Lists the files in the order in which they appear in the archive. Files may appear more than once.  |
| <b>u</b> | Adds <i>file</i> to the end of the archive only if it is not in the archive already or if it has been modified since it was written to the archive. Since the structure of a streaming tape device does not support the addition of information at the end of a tape, this is not a valid flag when the archival storage device is a streaming tape.  |
| <b>x</b> | Extracts <i>file</i> from the archive. If you specify a <i>directory</i> , <b>tar</b> extracts all files in that directory from the archive. If you do not specify a <i>file</i> or a <i>directory</i> , <b>tar</b> extracts all of the files from the archive. When an archive contains multiple copies of the same file, <b>tar</b> extracts only the last one and overwrites all earlier ones. If you have superuser authority (see “ <b>su</b> ” on page 1026), <b>tar</b> creates all files and directories with the same user and group IDs as on the tape. If you do not have superuser authority, the files and directories have your user and group IDs. |

The other optional flags to **tar** are listed below. In all cases, a directory parameter refers to all the files and subdirectories, recursively, within that directory. Flags without corresponding parameters may appear separately or be grouped together. Flags that take parameters may have them adjacent to the flag letter or as the entire following argument.

- |                 |  |
|-----------------|--|
| <b>-bblocks</b> | Specifies the number of 512-byte blocks per record. The default is 16, which is appropriate for tape records. Due to the size of inter-record gaps, tapes written with large blocking factors can hold much more data than tapes with only one block per record. |
|-----------------|--|

The block size is determined automatically when tapes are read (function flags **-x** or **-t**). When archives are updated with the **-u** and **-r** functions, the existing record size is used. The **tar** command writes archives using the specified *blocks* value only when creating new archives with the **-c** flag.

For output to ordinary files with the **-f** flag, you can save disk space by using a blocking factor that matches the size of disk blocks (for example, **-b4** for 2048-byte disk blocks). Ordinary files must be read using the same blocking factor used when they were created.

**-Cblocks**

Allows **tar** to use very large clusters of blocks when it deals with streaming tape archives. Note, however, that on input, **tar** cannot automatically determine the block size of tapes with very long block sizes created with this flag. In the absence of a **-Cnum** argument, the largest block size that **tar** can automatically determine is 20 blocks.

**-d**

Makes separate entries for directories, blocks and character special files, and FIFOs. Normally, **tar** writes only ordinary files to an archive, and extracts only ordinary files and the directories required to contain them as determined by the path names in the archive. When writing to an archive with the **-d** flag, **tar** makes it possible to preserve the directory permission codes and to restore empty directories, special files, and FIFOs with the **-x** flag.

**Note:** Although anyone can archive special files, only a user with superuser authority can extract them from an archive.

**-ffile[-num]**

Uses *file* as the archive to be read or written. When this flag is not specified, **tar** uses a system-dependent default file name of the form **/dev/rmt?**. If the *file* specified is **-** (minus), **tar** writes to standard output or reads from standard input. If you write to standard output, the **-c** flag must be used (see example).

If you specify *num*, **tar** provides automatic spill over from one archive storage unit to another. This feature allows the operator of a system with multiple tape drives to use multitape archives without having to change tapes. For example, **-f/dev/rmt0-2** writes or reads **/dev/rmt0**, followed by **/dev/rmt1**, and then **/dev/rmt2** before requesting that additional volumes be mounted.

**-h**

Ignores header checksum errors. The **tar** command writes a file header containing a checksum for each file in the archive. When this flag is not specified, the system verifies the contents of the header blocks by recomputing the checksum, and aborts with a **directory checksum error** when a mismatch occurs. When this flag is specified, **tar** logs the error and then scans forward until it finds a valid header block. This permits restoring files from later volumes of a multivolume archive without reading earlier volumes.

- 
- i***inputlist* Writes the files listed in the *inputlist* file to the archive. The *inputlist* should contain one file name per line. Files from *inputlist* are not treated recursively. If you include the name of a directory in *inputlist*, **tar** does not write that directory's subdirectories to the tape, only that directory's files. If you also list files or directories on the command line, the contents of *inputlist* are included after **tar** has written all the files or the directories and their subdirectories to the archive.
- l** Writes error messages to standard output if **tar** cannot resolve all of the links to the files archived. When you do not specify this flag, the system does not display these messages.
- m** Uses the time of extraction as the modification time. The default is to preserve the modification time of the files.
- s** *blocksb*  
**-s** *feet*  
**-s** *feet @density* Specifies the number of 512-byte *blocks* per volume (first format), independent of the tape blocking factor. You can also specify the size of the tape in feet by using the second form, and **tar** assumes a default *density*. The third form allows you to specify both tape length and density. Feet are assumed to be 11 inches long to be conservative. This flag lets you deal more easily with multivolume tape archives, where **tar** must be able to determine how many blocks fit on each volume.
- Note:** Tape drives vary in density capabilities. The *density* parameter calculates the amount of data a system can fit on a tape. This allows the correct amount of data to be written to a tape when using tape drives other than the IBM 6157 tape drive, which has a density of 700.
- v** Lists the name of each file as it is processed. With the **-t** flag, **-v** gives more information about the tape entries, including file sizes, times of last modification, UID, and GID, and permissions.
- w** Displays the action to be taken followed by the file name, then waits for user confirmation. If the response begins with a y or Y, then the action is performed. If the response is not affirmative, then the file is ignored.
- 

### Japanese Language Support Information

An affirmative response in Japanese Language Support matches one of the elements in the environment variable **YESSTR**.

---

## tar

---

`-num` Uses `/dev/rmtnum` instead of the default. For example, `-2` is the same as `-f/dev/rmt2`. In AIX systems with multidensity tape drives, this flag allows selecting a particular density. The default unit is system dependent and is chosen to match the default density, as described under the `-s` flag.

## Examples

1. To write **file1** and **file2** to a new archive on the default tape drive:  

```
tar -c file1 file2
```
2. To extract all files that are in the `/tmp` directory from the archive file on the tape device `/dev/rmt2` and use the time of extraction as the modification time:  

```
tar -xm -f/dev/rmt2 /tmp
```
3. To create a new archive file that contains **file1** and pass the archive file to the **dd** command to be written to the device `/dev/rmt1`:  

```
tar -cvf - file1 | dd of=/dev/rmt1
```
4. To display the names of the files in the disk archive file **out.tar** on the current directory:  

```
tar -vtf out.tar
```
5. To expand the compressed archive file **fil.tar.z**, pass the file to the **tar** command, and extract all files from the expanded archive file:  

```
pcat fil.tar.z | tar -xvf -
```

## Files

`/dev/rmt?`  
`/tmp/tar*`

## Related Information

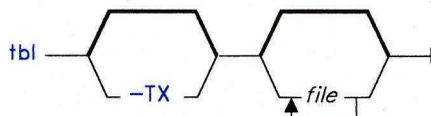
The discussion of Japanese Language Support in *Japanese Language Support User's Guide*.

## tbl

## Purpose

Formats tables for the **nroff**, **troff** and **troff** commands.

## Syntax



OL805222

## Description

The **tbl** command is a preprocessor that formats tables for **nroff** and **troff**. It reads the specified *files* or, if you do not specify any file names or you specify a - (minus) as a file name, it reads standard input. The input is copied unchanged to standard output, except for text between lines containing **.TS** and **.TE**. This text describes tables, and is reformatted by **tbl**. The **.TS** and **.TE** lines are not altered by **tbl**. For more detailed information on how to format text for **tbl**, see *Text Formatting Guide*.

**Note:** When **tbl** is used with **eqn** or **neqn**, **tbl** should come first to minimize the volume of data passed through pipelines.

At the start of **tbl** text, you should include a line containing **.TS**. You can follow this with a line containing global options. The available global options are:

<b>center</b>	Centers the table (the default is left-adjusted).
<b>expand</b>	Makes the table as wide as the current line length.
<b>box</b>	Encloses the table in a box.
<b>doublebox</b>	Encloses the table in a double box.
<b>allbox</b>	Encloses each item of the table in a box.
<b>tab (x)</b>	Uses the character <i>x</i> instead of a tab to separate items in a line of input data.
<b>linesize (n)</b>	Sets lines and rules for boxes in point size <i>n</i> .
<b>delim (x,y)</b>	Sets <i>x</i> and <i>y</i> as the <b>eqn</b> and <b>neqn</b> text delimiters.

End the list of global options with a ; (semicolon).

After the global options, enter lines describing the format of each row in the table. Each format line (except the last) describes one row of the table. The last one describes all remaining rows of the table. This must end with a period to indicate that it is the end of the format specification. Each column of the table is described by a single keyletter.

The available keyletters are:

- c** Centers the item in the column.
- r** Right-adjusts the item in the column.
- l** Left-adjusts the item in the column.
- n** Adjusts the numerical items in the column to line up at the decimal point or right-adjusts them if there are no decimal points.
- s** Allows the previous item on the left to spill over into this column if the item is too wide for its column.
- a** Centers the longest line in this column and then left-adjusts all other lines in it with respect to the centered line.
- ^** Allows the item above to spill over into this column if the item is too large.
- Replaces this entry with a horizontal line.
- =** Replaces this entry with a double horizontal line.

After the keyletter, you can enter specifiers that determine where vertical lines appear between columns, column width, inter-column spacing, and the font and point size of the item. The following table lists the legal specifiers.

Specifier	Meaning	Specifier	Meaning
<b>e</b> or <b>E</b>	Equal width columns	<b>w</b> or <b>W</b>	Minimum width column
<b>f</b> or <b>F</b>	Font change	<b>z</b> or <b>Z</b>	Zero width column
<i>nnn</i>	Column separation	<i>.xx</i>	Included <b>troff</b> request
<b>p</b> or <b>P</b>	Point size change		Vertical line
<b>s</b> or <b>S</b>	Spanned item		Double vertical line
<b>t</b> or <b>T</b>	Vertical spanning at top	\^	Vertical span
<b>T</b> { . . . <b>T</b> }	Text block	\-	Short horizontal line
<b>u</b> or <b>U</b>	Staggered columns	\R <i>x</i>	Repeat character
<b>v</b> or <b>V</b>	Vertical spacing change		

**Figure 11.** tbl Column and Item Specifiers

The format lines are followed by lines containing data for the table. The last line consists of **.TE**. Within the data lines, data items are separated by tab characters, unless the global option, **delim** is used.

If a data line consists of only **-** (underscore) or **=** (equal sign), a single or double line is drawn across the table at that point. If an entry in a data line consists of only **-** or **=**, then that item is replaced by a single or double line.

## Flag

- TX** Uses only full vertical line motions, making the output suitable for line printers and other devices that do not have partial vertical line motions.

## Related Information

The following commands: “**cw, checkcw**” on page 275, “**eqn, neqn, checkeq**” on page 395, “**mm, checkmm**” on page 663, “**mmt, checkmm**” on page 666, “**nroff, troff**” on page 709, and “**troff**” on page 710.

The **mm** and **mv** miscellaneous facilities in *AIX Operating System Technical Reference*.

The discussion of **tbl** in *Text Formatting Guide*.

tc

---

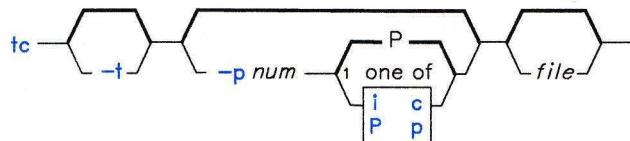
tc

---

## Purpose

Simulates phototypesetter output for a Tektronix 4014 work station.

## Syntax



OL805271

<sup>1</sup> Do not put a blank between these items.

OL805308

## Description

The `tc` command interprets its input, either a *file* or standard input, as a **troff** document. It then simulates the typesetter output for a Tektronix 4014 work station with ASCII and APL character sets and writes the results to standard output (usually the work station display). The 16 typesetter sizes are mapped into the 4014's four sizes; the entire **troff** character set is drawn using the 4014's character generator, with overstruck combinations where necessary.

At the end of each page, `tc` waits for a new-line character from the keyboard before continuing to the next page. While it is waiting, the command `e` suppresses the screen erase before the next page. `!AIX-cmd` sends `AIX-cmd` to the shell.

There are no font distinctions in the display.

## Flags

- `-pnum letter` Sets page length to *num* and scale to *letter*. *letter* may include the scale factors **p** (points), **i** (inches), **c** (centimeters), and **P** (picas). The default is picas. Do not put a space between *num* and *letter*.
- `-t` Does not wait between pages (use in a pipeline).

## Example

To use **tc** in a pipeline with **troff**:

```
troff -t chapter1 | tc
```

## Related Information

The following commands: “**sh**” on page 913, “**tplot**” on page 1079, “**troff**” on page 710, and “**4014**” on page 1264.

# tctl

---

## tctl

---

### Purpose

Gives commands to streaming tape.

### Syntax

```
tctl [-f$TAPE] [-f tapename] subcmd [count]
```

OL805397

### Description

The **tctl** command gives subcommands to a streaming tape device. If you do not specify the **-f** flag with *tapename*, the environment variable **TAPE** is used. If the environment variable does not exist, **tctl** uses the device **/dev/rmt4**. The *tapename* parameter must be a raw (not block) tape device. You can specify more than one operation with *count*.

### Subcommands

- eof**
- weof**      Writes *count* end-of-file markers at the current position on the tape.
- fsf**        Moves the tape forward *count* files.
- fsr**        Moves the tape forward *count* records.
- rewind**     Rewinds the tape. The *count* parameter is ignored.  
**Note:** It is sometimes necessary to issue a **reset** before issuing a **rewind** subcommand.
- offline**
- rewoffl**
- reset**      Places the tape drive off-line. The *count* parameter is ignored.
- erase**      Erases all contents on the tape and rewinds it.
- retension**   Moves the tape to the beginning, the end, and back to the beginning of the tape. If you have excessive read errors during a restore operation, you should run the **retension** subcommand. If the tape has been exposed to environmental extremes, you should run the **retension** subcommand before the save operation.

- ras1** Performs a checksum on the tape drive.
- ras2** Checks the capstan speed, verifies the operations of the BOT, EOT, and SAFE sensors, and writes a worst case pattern on the tape and attempts to verify the pattern.

## Files

/dev/rmt?? The raw streaming tape interface.

## Related Information

The following command: “**dd**” on page 301.

The **ioctl** system call and the **tape** and **environ** files in *AIX Operating System Technical Reference*.

tee

---

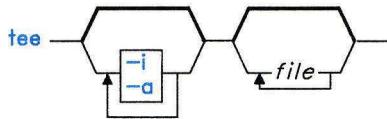
tee

---

## Purpose

Displays the output of a program and copies it into a file.

## Syntax



OL805272

## Description

The **tee** command reads standard input and writes the output of a program to standard output and copies it into *file* at the same time.

## Flags

- a Adds the output to the end of *file* instead of writing over it.
- i Ignores interrupts.

**Note:** If you specify both flags, each must appear separately on the command line, preceded by a - (minus).

## Examples

1. To view and save the output from a command at the same time:

```
lint program.c | tee program.lint
```

This displays the standard output of the command `lint program.c` at the work station, and at the same time saves a copy of it in the file `program.lint`. If `program.lint` already exists, it is deleted and replaced.

2. To display and append to a file:

```
lint program.c | tee -a program.lint
```

This displays the standard output of `lint program.c` at the work station and at the same time appends a copy of it to the end of `program.lint`. If the file `program.lint` does not exist, it is created.

# termdef

---

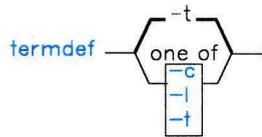
## termdef

---

### Purpose

Queries terminal characteristics.

### Syntax



OL805454

### Description

The **termdef** command identifies the current display type, the active lines setting, or the current columns setting, thus simplifying the task of resetting the lines and columns when you switch fonts or of resetting the **\$TERM** environment variable when you switch displays. The **terminfo** file defines the default number of lines and columns for each display, but the lines and columns can change depending upon which font is currently active. In addition, the **\$TERM** environment variable does not automatically reflect the display currently being used. If you are using a display other than the **ibm5151**, you must explicitly reset this variable to access the **terminfo** correctly.

### Flags

- c Returns the current column value.
- l Returns the current lines value.
- t Returns the name of the current display (this is the default action).

### Example

To set environment variables according to the values of the currently active font and display, add the following lines to the **/etc/rc** file:

```
TERM=`termdef`  
COLUMNS=`termdef -c`  
LINES=`termdef -l`  
export TERM LINES COLUMNS
```

## **Related Information**

The following command: “**display**” on page 332.

The **terminfo** file and the **hft** special file in *AIX Operating System Technical Reference*.

## test

---

## test

---

### Purpose

Evaluates conditional expressions.

### Syntax

```
test — expression —  
[ — expression — ] —
```

OL805273

### Description

The **test** command evaluates *expression* and, if its value is true, returns a zero (true) exit value. otherwise it returns a nonzero (false) exit value; **test** also returns a nonzero exit value if there are no parameters.

**Note:** In the second form of the command, that is the one that uses square brackets ([ ]), rather than the word **test**, the brackets must be surrounded by blanks.

### Functions

All the functions and operators are separate parameters to **test**. The following functions are used to construct *expression*:

<b>-r</b> <i>file</i>	True if <i>file</i> exists and has read permission.
<b>-w</b> <i>file</i>	True if <i>file</i> exists and has write permission.
<b>-x</b> <i>file</i>	True if <i>file</i> exists and has execute permission.
<b>-f</b> <i>file</i>	True if <i>file</i> exists and is a regular file.
<b>-d</b> <i>file</i>	True if <i>file</i> exists and is a directory.
<b>-c</b> <i>file</i>	True if <i>file</i> exists and is a character special file.
<b>-b</b> <i>file</i>	True if <i>file</i> exists and is a block special file.
<b>-p</b> <i>file</i>	True if <i>file</i> exists and is a named pipe (FIFO).
<b>-u</b> <i>file</i>	True if <i>file</i> exists and its set-user-ID bit is set.

---

<b>-g</b> <i>file</i>	True if <i>file</i> exists and its set-group-ID bit is set.
<b>-k</b> <i>file</i>	True if <i>file</i> exists and its <b>sticky bit</b> is set.
<b>-s</b> <i>file</i>	True if <i>file</i> exists and has a size greater than zero.
<b>-t</b> [ <i>filedescr</i> ]	True if the open file with file descriptor number <i>filedescr</i> (1 by default) is associated with a work station device.
<b>-z</b> <i>s1</i>	True if the length of string <i>s1</i> is zero.
<b>-n</b> <i>s1</i>	True if the length of the string <i>s1</i> is nonzero.
<i>s1</i> = <i>s2</i>	True if strings <i>s1</i> and <i>s2</i> are identical.
<i>s1</i> != <i>s2</i>	True if strings <i>s1</i> and <i>s2</i> are not identical.
<i>s1</i>	True if <i>s1</i> is not the null string.
<i>n1</i> <b>-eq</b> <i>n2</i>	True if the integers <i>n1</i> and <i>n2</i> are algebraically equal. Any of the comparisons <b>-ne</b> , <b>-gt</b> , <b>-ge</b> , <b>-lt</b> , and <b>-le</b> can be used in place of <b>-eq</b> .

These functions can be combined with the following operators:

<b>!</b>	Unary negation operator.
<b>-a</b>	Binary AND operator.
<b>-o</b>	Binary OR operator ( <b>-a</b> has higher precedence than <b>-o</b> ).
<b>\( expression \)</b>	Parentheses for grouping.

## Examples

- To test whether a file exists and is not empty:

```
if test ! -s "$1"
then
    echo $1 does not exist or is empty.
fi
```

If the file specified by the first positional parameter to the shell procedure does not exist, this displays an error message. If \$1 exists, it displays nothing. Note that there must be a space between **-s** and the file name.

The double quotation marks around \$1 ensure that the test will work properly even if the value of \$1 is the empty string. If the double quotation marks are omitted and \$1 is the empty string, **test** displays the error message `test: parameter expected`.

2. To do a complex comparison:

```
if [ $# -lt 2 -o ! -s "$1" ]
then
    exit
fi
```

If the shell procedure was given fewer than two positional parameters or the file specified by \$1 does not exist, then this exits the shell procedure. The special shell variable \$# represents the number of positional parameters entered on the command line that started this shell procedure. For more details, see “Shell Variables and Command-Line Substitutions” on page 917.

### Related Information

The following commands: “**find**” on page 422 and “**sh**” on page 913.

---

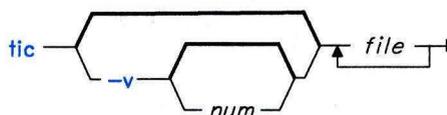
# tic

---

## Purpose

Translates **terminfo** files from source to compiled format.

## Syntax



OL805340

## Description

The **tic** command translates **terminfo** files from the source format into the compiled format. **tic** places the results in the directory **/usr/lib/terminfo**. If the environment variable **TERMINFO** is set, the results are placed there instead of in **/usr/lib/terminfo**.

The **tic** command compiles all terminfo descriptions in *files*. When **tic** finds a **use =** field, it searches first the current file, then the master file, **./terminfo.src**.

The total compiled entries cannot exceed 4096 bytes and the name field cannot exceed 128 bytes.

## Flag

**-vnum** Writes trace information on the progress of **tic**. *num* is an integer that increases the level of the verbosity.

## Files

**/usr/lib/terminfo/?/\*** Compiled terminal capability database.

## Related Information

The **curses** subroutine and the **terminfo** file in *AIX Operating System Technical Reference*.

# time

---

## time

---

### Purpose

Times the execution of a command.

### Syntax

```
time — command —
```

OL805274

### Description

The **time** command times the execution of the named *command*. **time** writes to standard error the elapsed time of the command, the system time used, and the execution time, in seconds.

### Examples

1. To measure the time required to run a program:

```
time a.out
```

This runs the program **a.out** and writes to the standard error output the amount of real, system, and user time that it uses:

```
real      10.5
user       0.3
sys        3.6
```

2. To save a record of the **time** information in a file:

```
time a.out 2> a.time
```

### Related Information

The following commands: “**cs**h” on page 225 and “**tim**ex” on page 1069.

**Note:** The **cs**h command contains a built-in subcommand named **time**. The command and subcommand do not necessarily work the same way. For information on the subcommand, see the **cs**h command.

The **times** system call in *AIX Operating System Technical Reference*.

---

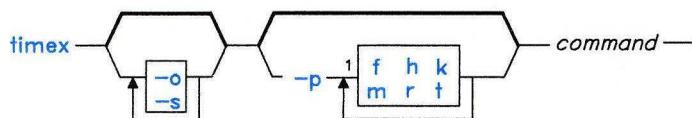
# timex

---

## Purpose

Times a command, and reports process data and system activity.

## Syntax



<sup>1</sup>Do not put a blank between these items.

OL805275

## Description

The **timex** command reports, in seconds, the elapsed time, user time, and system execution time for *command* where *command* is a local command. With flags specified, **timex** can list or summarize process accounting data for *command* and all of its children, and report total system activity during the execution interval. Output is written to standard error. The system uses the `/usr/adm/pacct` to select process records associated with *command* and includes background processes having the same user ID, work station ID, and execution time window.

## Flags

- o Reports the total number of blocks read or written and total characters transferred by *command* and all its children.
- p Lists process accounting records for *command* and all its children. The number of blocks read or written and the number of characters transferred are always reported. The **f**, **h**, **k**, **m**, **r**, and **t** arguments, defined in the **acctcom** command, modify the other data items reported.
- s Reports total system activity that occurred during the execution of *command*. All the data items listed in the **sar** command are reported.

**timex**

---

## **Related Information**

The following commands: **“acctcom”** on page 20 and **“sar”** on page 867.



# tlogger

---

## tlogger

---

### Purpose

Gathers I/O from a terminal and writes it to a log file.

### Syntax

```
tlogger -c /usr/adm/ras/tlogfile -b /usr/adm/ras/tlogfile.bk &
```

*-c file*      *-b file*

AJ2FL105

### Description

The terminal-logging daemon **tlogger** collects data read or written to its associated terminal and writes that data to a log file. Each time the daemon is started, the contents of the current log file replace the backup log file. A new current log file is created with permissions set to allow read and write by the owner.

The associated terminal is identified in the following manner: Standard error is assumed to be the correct terminal if it is a terminal device (**isatty()** returns true). Otherwise, the process's **usrinfo** is used to identify the login terminal, and that device is used.

The **tlogger** daemon creates a message queue, and passes that queue ID to the associated terminal using the **ioctl TCLOG** system call. The daemon then loops waiting on message queue data; it writes any message queue data it receives to the end of the current log file. The terminal log daemon will catch all signals (except **SIGKILL**). On receipt of a signal, the daemon issues an **ioctl TCLOG** to its associated terminal to turn off logging. This causes the terminal to stop sending log messages. The daemon then removes the message queue and exits. The daemon also terminates if it can no longer write to the log file due to file size constraints. In this case, an error message is written to standard error.

The **tlogger** daemon should be started in the background either from **/etc/rc**, or from the command line. This starts the terminal sending its I/O data to the daemon. The **tlog** command can then be used to stop or restart the sending of terminal I/O to the daemon. The daemon itself may be terminated with the **kill** command, but would ordinarily continue to run until shutdown occurs.

## Notes:

1. **SIGKILL** should not be used to stop the daemon, since cleanup of system resources cannot be done in that case.
2. It may be necessary to prevent passwords from showing up in the terminal logs. You can prevent the system from logging passwords by having the **getpass()** subroutine turn off terminal logging while it is reading the password. The **login**, **adduser**, **newgrp**, and **passwd** commands use this subroutine.

## Flags

- b filename** Specifies a file to be used as the backup log file. The default backup file is **/usr/adm/ras/tlogfile.bk**.
- c filename** Specifies a file to be used as the current log file. The default current file is **/usr/adm/ras/tlogfile**.

## Files

- /etc/rc** System startup file.
- /usr/adm/ras/tlogfile** Default current log file.
- /usr/adm/ras/tlogfile.bk** Default backup log file.

## Related Information

The following commands: “**tlog**” on page 1071, “**shutdown**” on page 946, and “**kill**” on page 552.

The **ioctl** system call and the **getpass** subroutine in *AIX Operating System Technical Reference*.

toc

---

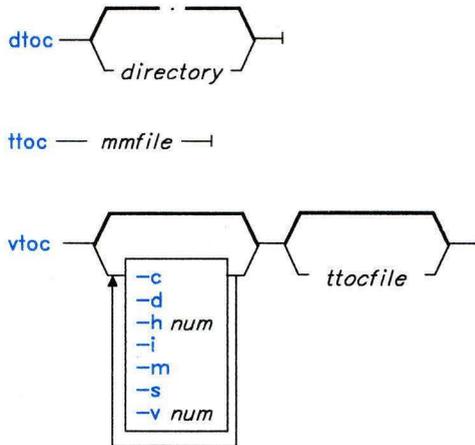
toc

---

## Purpose

Provides graphical table of contents routines.

## Syntax



OL777076

## Description

All of the commands listed below reside in `/usr/bin/graf` (see “**graphics**” on page 497).

### dtoc

The **dtoc** command makes a textual table of contents, **TTOC**, of all subdirectories beginning at *directory* (by default the current directory.). The list has one entry per directory. The entry fields from left to right are level number, directory name, and the number of ordinary readable files in the directory. **dtoc** is useful in making a visual display of all or parts of a file system. The following will make a visual display of all the readable directories under the root directory (/):

```
dtoc / | vtoc | td
```

**ttoc**

Output is the table of contents generated by the `.tc` macro of the `mm` command translated to **TTOC** format. The input is assumed to be a `mm` file that uses the `.H` family of macros for section headers. If no *file* is given, the standard input is assumed.

**vtoc**

The `vtoc` command produces a **GPS** describing a hierarchy chart from a **TTOC**. The output drawing consists of boxes containing text connected in a tree structure. If no *file* is given, the standard input is assumed. Each **TTOC** entry describes one box and has the form:

*id*[*line-weight,line-style*]"*text*"[*mark*]

where:

*id* is an alternating sequence of numbers and dots. The *id* specifies the position of the entry in the hierarchy. The *id* `0` is the root of the tree.

*line-weight* is either:

**n**, normal-weight; or  
**m**, medium-weight; or  
**b**, bold-weight.

*line-style* is either:

**so**, solid-line;  
**do**, dotted-line;  
**dd**, dot-dash line;  
**da**, dashed-line; or  
**ld**, long-dashed

*text* is a character string surrounded by quotes. The characters between the quotes become the contents of the box. To include a quote within a box, it must be escaped (`\`).

*mark* is a character string (surrounded by quotes if it contains spaces). To include a dot within a box, it must be escaped (`\.`). The string is put above the top right corner of the box. To include either a quote or a dot within a *mark* it must be escaped.

Entry example:

```
1.1b,da"ABD" DEF
```

Entries may span more than one line by escaping the new-line (`\new-line`).

Comments are surrounded by the `/*,*` pair. They can appear anywhere in a **TTOC**.

### ***Flags***

- c** Uses text as entered (default is all upper case).
- d** Connects the boxes with diagonal lines.
- hnum** Sets horizontal interbox space to *num*% of box width.
- i** Suppresses the box *id*.
- m** Suppresses the box *mark*.
- s** Do not compact boxes horizontally.
- vnum** Vertical interbox space is *num*% of box height.

### **Related Information**

The following command: “**graphics**” on page 497.

The **gps** file in *AIX Operating System Technical Reference*.

---

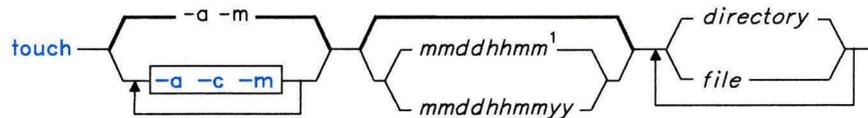
# touch

---

## Purpose

Updates the access and modification times of a file.

## Syntax



<sup>1</sup>The current year is the default year.

OL805276

## Description

The **touch** command updates the access and modification times of each *file* or *directory* named to the one specified on the command line. If you do not specify a time, **touch** uses the current time. If you specify a file that does not exist, **touch** creates a file with that name unless you request otherwise with the **-c** flag.

The environment variables **NLDATE** and **NLTIME**, if defined, specify the order of month and day in the date specification and of hour and minute in the time specification. Otherwise, these orders default to *mmdd* and *hhmm*.

The return code from **touch** is the number of files for which the times could not be successfully modified (including files that did not exist and were not created).

## Flags

- a** Changes only the access time.
- c** Does not create the file if it does not already exist.
- m** Changes only the modification time.

# touch

---

## Examples

1. To update the access and modification times of a file:

```
touch program.c
```

This sets the last access and last modification times of `program.c` to the current date and time. If `program.c` does not exist, **touch** creates an empty file with that name.

2. To avoid creating a new file:

```
touch -c program.c
```

3. To update only the modification time:

```
touch -m *.o
```

This updates only the last modification times of the files in the current directory that end with `.o`. **touch** is often used in this way to alter the results of the **make** command.

4. To explicitly set the access and modification times:

```
touch -c 02171425 program.c
```

This sets the access and modification dates to 14:25 (2:25 p.m.) February 17 of the current year.

## Related Information

The following command: “**date**” on page 281.

The **utime** system call in *AIX Operating System Technical Reference*.

“Overview of International Character Support” in *Managing the AIX Operating System*.

---

# tplot

---

## Purpose

Produces plotting instructions for a particular work station.

## Syntax

```
tplot -T$TERM -Tworkstation file
```

OL805277

## Description

The **tplot** command reads plotting instructions from standard input or from *file*, if specified. (For more information about plotting instructions, see the **plot** file format *AIX Operating System Technical Reference*). **tplot** writes instructions suitable for the specified *workstation* to standard output. If *workstation* is not specified, the environment variable **TERM** is used. (For more information about environment variables, see the **environ** file in *AIX Operating System Technical Reference*).

---

### Japanese Language Support Information

This command has not been modified to support Japanese characters.

---

## Flag

**-Tworkstation** Uses the plotting instructions for *workstation*. The known *workstation* is:

<b>lp</b>	IBM PC graphics printer
-----------	-------------------------

## Files

/usr/lib/tcolor  
/usr/lib/tprint

## **tplot**

---

### **Related Information**

The following commands: “**graph**” on page 494 and “**splp**” on page 975.

The **plot** subroutine and the **plot** file in *AIX Operating System Technical Reference*.

---

# tput

---

## Purpose

Queries the **terminfo** file.

## Syntax

```
tput -T$TERM -Ttype cmd capname
```

OL805398

## Description

The **tput.** command uses the **terminfo** file to make terminal-dependent information available to the shell. The output of **tput.** is a string if the attribute *capname* (for capability name) is of type string or an integer if the attribute is of type integer. If the attribute is of type Boolean, **tput.** simply sets the exit value (0 for TRUE, 1 for FALSE), and produces no other output.

## Flags

- Ttype** Indicates the type of work station. Normally, the value of *type* is supplied by the environment variable **\$TERM**.
- capname* Indicates the attribute from the **terminfo** file. For more information, see the **terminfo** file in *AIX Operating System Technical Reference*.

## Examples

1. To echo the clear-screen sequence for the current work station:  
`tput clear`
2. To display the number of columns for the current work station:  
`tput cols`
3. To display the number of columns for the 450 work station:  
`tput -T450 cols`

## tput

---

4. To set the shell variable `bold` to the highlight mode sequence for the current work station:

```
bold=`tput smso`
```

This might be followed by a prompt:

```
echo "${bold}Please type in your name: \c"
```

5. To set the exit value to indicate if the current work station is a hardcopy terminal:

```
tput hc
```

## Files

<code>/usr/lib/terminfo/?/*</code>	Terminal descriptor files.
<code>/usr/include/term.h</code>	Definition files.
<code>/usr/include/curses.h</code>	

## Related Information

The following command: “`stty`” on page 1018.

The `terminfo` file in *AIX Operating System Technical Reference*.

---

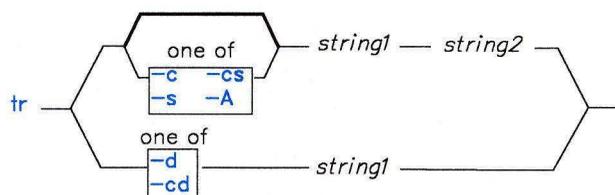
**tr**


---

## Purpose

Translates characters.

## Syntax



OL805278

## Description

The **tr** command copies characters from the standard input to the standard output with substitution or deletion of selected characters. Input characters from *string1* are replaced with the corresponding characters in *string2*. **tr** cannot handle an ASCII NUL (`\000`) in *string1* or *string2*; it always deletes NUL from the input.

Abbreviations that can be used to introduce ranges of characters or repeated characters are:

- [a-z]**           Stands for a string of characters whose ASCII codes run from character **a** to character **z**, inclusive.
- [a\*num]**       Stands for *num* repetitions of **a**. *num* is considered to be in decimal unless the first digit of *num* is **0**; then it is considered to be in octal.

Use the escape character `\` (backslash) to remove special meaning from any character in a string. Use the `\` followed by 1, 2, or 3 octal digits for the ASCII code of a character.

---

### Japanese Language Support Information

You can use two octal sequences to specify a 2-byte kanji character. If you specify ranges of kanji characters, they are interpreted for translation as a string of kanji characters in ascending sequence in their binary representation.

---

## Flags

- A Translates on a byte-by-byte basis. When you specify this flag, **tr** does not support extended characters.
- c Complements (inverts) the set of characters in *string1* with respect to the universe of characters whose ASCII codes are 001 through 377 octal, if you specify -A, and all characters, if you do not specify -A.
- d Deletes all input characters in *string1*.
- s Changes characters that are repeated output characters in *string2* into single characters.

## Examples

1. To translate braces into parentheses:

```
tr '{} '()' <textfile >newfile
```

This translates each { to ( and each } to ). All other characters remain unchanged.

2. To translate lowercase characters to uppercase:

```
tr '[a-z]' '[A-Z]' <textfile >newfile
```

3. This is what happens if the strings are not the same length:

```
tr '[0-9]' '#' <textfile >newfile
```

This translates each 0 to a # (number sign).

**Note:** If the two character strings are not the same length, then the extra characters in the longer one are ignored.

4. To translate each digit to a #:

```
tr '[0-9]' '[#*]' <textfile >newfile
```

The \* tells **tr** to repeat the # enough times to make the second string as long as the first one.

5. To translate each string of digits to a single *num*:

```
tr -s '[0-9]' '[#*]' <textfile >newfile
```

6. To translate all ASCII characters that are *not* specified:

```
tr -c '[-~]' '[A-~]?' <textfile >newfile
```

This translates each nonprinting ASCII character to the corresponding control key letter (\001 translates to A, \002 to B, etc.). ASCII DEL (\177), the character that follows ~ (tilde), translates to ?.

7. To create a list of the words in a file:

```
tr -cs '[a-z][A-Z]' '[\012*]' <textfile >newfile
```

This translates each string of nonalphabetic characters to a single new-line character. The result is a list of all the words in `textfile`, one word per line.

## Related Information

The following commands: “**ed**” on page 371 and “**sh**” on page 913.

The `ascii` file in *AIX Operating System Technical Reference*.

“Overview of International Character Support” in *Managing the AIX Operating System*.

The discussion of Japanese Language Support in *Japanese Language Support User’s Guide*.

## trace

---

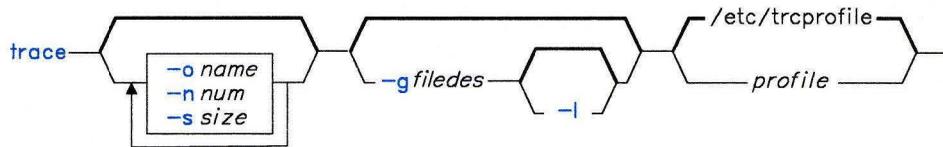
## trace

---

### Purpose

Starts the trace function.

### Syntax



OL805279

### Description

The **trace** command starts the trace function in the background. This trace function provides a base for debugging the system. **trace** monitors the occurrence of selected events in the system and records on disk important data specific to each of these events. You can format this output with the **trcrpt** command.

Any user or program that needs the trace process enabled for debugging or error determination can start **trace**. When starting **trace**, you must provide a *profile*. This allows you to tailor the output of the trace session to individual needs. The default *profile* is `/etc/trcprofile`.

There may be more than one trace profile in the file system at a time. The trace profile contains the classes of events that you can select to trace, listed by event class and by a descriptive label. See “Example” on page 1089 for a sample profile. You may keep different profiles to trace different combinations of event classes. **trace** also takes additional information about the trace session from the configuration file `/etc/rasconf` (see *AIX Operating System Technical Reference* for a discussion of this file). You set the name and size of the output file in this configuration file.

In a multiuser environment, **trace** records all system events, not just events at one virtual terminal.

---

## Japanese Language Support Information

This command has not been modified to support Japanese characters.

---

## Flags

**-g** *filedes* Indicates that this is a **generic** trace session. Generic tracing applies only to the VRM. In this type of session, events to be recorded do not necessarily have a fixed event class, but are allocated to a temporary event channel by the trace device driver, **/dev/vrmtrace**. Thus, starting a generic trace does not require a trace profile. Generic traces are started and stopped by other processes, such as communications session managers. Therefore, the interface to the daemon is somewhat different. The **-g** flag is useful only when **trace** is started by another process.

The *filedes* parameter is a file descriptor from the parent process. **trace** writes this following information to this file descriptor:

- The process ID of the **trace** demon
- The address of the trace buffer
- The size (in bytes) of the trace buffer
- The temporary channel bit allocated to this event.

When tracing a generic event, the **trace** demon does not record its process ID so that it can be stopped by the **trcstop** command. Thus, more than one **trace** demon may be running at any time, but there may be as many as seven traces in the system at once (one normal trace and from one to six generic traces).

Use the **trc\_start** and **trc\_stop** subroutines to start and stop a generic trace.

**-l** Indicates that the VRM trace device driver should log only the last buffer filled before the **trace** demon stops. This flag is valid only during a generic trace (**-g**).

**-o** *name* Specifies the name of the log file into which the **trace** demon stores the trace data. For generic traces (**-g**), this name must be different from the default file name specified in the configuration file **/etc/rasconf**.

**-n** *num* Specifies the number of entries in the trace buffer. **trace** multiplies this number by the size of the entries (see the **-s** flag) and uses the resulting value to size the trace buffer. If you do not specify this flag, **trace** uses the buffer size specified in the configuration file **/etc/rasconf**.

## trace

---

**-s** *size* Specifies the size (in bytes) of the entries that the **trace** demon will be handling. The default size is 40 bytes. The size can be no less than 20, which is the number of bytes in the trace header for each entry. All entries must be the same size in a particular trace log file.

**Example**

```

*****
* SYSTEM TRACE PROFILE
*****
* To set trace on for an event class, remove the comment mark (*) from the
* first column of the line containing the event you wish to trace.
* Add a comment mark (*) in the first column of lines containing event type
* you wish to stop tracing.

***** Event
*      Type      Description

*****
                Applications

*****
                Kernel Extensions
*      36        Config

*****
                Kernel System Calls
*      60        Shared Memory
*      61        Messages
*      62        Semaphores
*      63        Signals
*      64        Time
*      65        File System
*      66        File Handling
*      67        Directory Handling
*      68        Process

*****
                VRM Components
*      100       SVC Handler
*      110       Async/5080 Peripherals
*      112       Async/5080 Peripheral Interrupts
*      113       Virtual Terminal Manager
*      114       Keyboard Interrupts
*      115       Locator Interrupts

*      150       User-Defined Events

```

## trace

---

### Files

/etc/trcprofile	Default profile.
/usr/adm/ras/trcfile	Output file defined in <b>/etc/rasconf</b> .
/etc/rasconf	Configuration file.

### Related Information

The following commands: “**trcstop**” on page 1093 and “**trcrpt**” on page 1091.

The **rasconf** configuration file in *AIX Operating System Technical Reference*.

The discussion of **trace** in *AIX Operating System Programming Tools and Interfaces*.

---

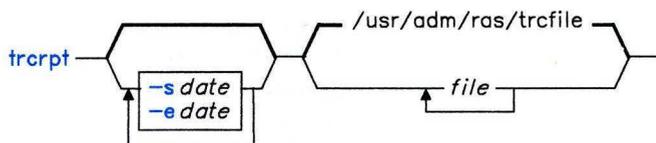
# trcrpt

---

## Purpose

Formats a report from the trace log file.

## Syntax



OL805280

## Description

The **trcrpt** command writes to standard output a chronological listing in readable format of the trace log *file* or *files* specified. You can specify a maximum of ten log files. If you do not specify any files, **trcrpt** reads */etc/rasconf* for a file name. This name is usually */usr/adm/ras/trcfile*.

---

### Japanese Language Support Information

This command has not been modified to support Japanese characters.

---

## Flags

- e date** Ends the report time with entries on or before *date*. The format of *date* is the same as the **date** command, *MMddhhmmyy*.
- s date** Starts the report with entries on or later than *date*. The format of *date* is the same as the **date** command, *MMddhhmmyy*. If you do not specify this flag, **trcrpt** formats the entire log file.

## Example

To format a trace log file:

```
trcrpt -s0109100384 -e0109100584 /u/dave/trc_log | print
```

## trcrpt

---

This formats the log file /u/dave/trc\_log, starting with entries from January 09, 1984 at 10:03 and ending at 10:05. It pipes the formatted output to the print queue.

### Files

/usr/adm/ras/trcfile	Default log file.
/etc/trcfmt	Trace format file.
/usr/adm/ras/.trcevents	Trace event types table.

### Related Information

The following commands: “**trace**” on page 1086 and “**trcstop**” on page 1093.

The **rasconf** file in *AIX Operating System Technical Reference*.

The discussion of **trcrpt** in *AIX Operating System Programming Tools and Interfaces*.

# **trcstop**

---

## **Purpose**

Stops the trace function.

## **Syntax**

`trcstop` —|

OL805223

## **Description**

The **trcstop** command sends a Software Terminate signal to the **trace** background process. This gracefully ends **trace** and forces cleanup.

---

### **Japanese Language Support Information**

This command has not been modified to support Japanese characters.

---

## **Files**

`/tmp/trc_PIDs`

## **Related Information**

The following commands: “**trace**” on page 1086 and “**trcrpt**” on page 1091.

The discussion of **trcstop** in *AIX Operating System Programming Tools and Interfaces*.

# trcupdate

---

## trcupdate

---

### Purpose

Updates trace format templates.

### Syntax

`trcupdate` *file* 

OL805399

### Description

The **trcupdate** command adds, replaces, or deletes trace report format templates in the files `/etc/trcfmt` and `/usr/adm/ras/.trcevents` and event types in the file `/etc/trcprofile`. **trcupdate** creates three undo files in the current directory named `file.undo.trc`, `.trcevents.undo.evt`, and `file.undo.pro`. These undo files can be used as input to **trcupdate** with the **-o** (override) flag to undo the changes **trcupdate** has just made.

The **trcupdate** command reads three files named `file.trc`, `file.evt`, and `file.pro`. The **trc** file contains trace format templates; the **evt** file contains trace event types and their corresponding hook IDs; the **pro** file contains the event type line for the trace profile.

The first field of each template contains an operator:

- + To add or replace a template
- To delete a template.

If the operation is **+**, then the following fields contain the template to be replaced. The hook ID of the template is also added to the `/usr/adm/ras/.trcevents` file, and the event type line is added to the trace profile `/etc/trcprofile`. If the operation is a **-**, then the second field contains the hook ID of the template to delete. That hook ID is also deleted in `/usr/adm/ras/.trcevents`, and the event type line is deleted from `/etc/trcprofile`.

When adding or replacing, **trcupdate** compares the version numbers of each input template with the version number of the existing template of the same hook IDs. If the version number of the input template is later, it replaces the old template with the input template. If the template does not already exist, then it is added to the file. The input file *must* contain the identifier `* /etc/trcfmt` on the first line.

The *file.evt* file contains a table of trace system event types and hook IDs that fall under these types. **trcupdate** reads in the file */usr/adm/ras/.trcevents* and adds in any hook IDs from *file.evt* that are not already accounted for or reassigns/deletes hook IDs to the event type given in the update file. The first line of the event/hook update file *must* be:

\* */ras/.trcevents* or **trcupdate** rejects the input file.

The *file.pro* contains the lines that are to be added to or deleted from */etc/trcprofile*. **trcupdate** reads */etc/trcprofile* and adds or deletes the specified event type line from *file.pro*. The first line of the event type file *must* be: \* */etc/trcprofile* or **trcupdate** rejects the input file.

---

### Japanese Language Support Information

This command has not been modified to support Japanese characters.

---

## Flag

-o Does no version number checking.

## Examples

1. The following is a sample **trc** file:

```
* /etc/trcfmt
+ 355 1.0 new_fmt
- 351
- 352
```

2. The following is a sample **evt** file:

```
* ras/.trcevents
350 355 356 357
```

## Files

```
/etc/trcfmt
/usr/adm/ras/.trcevents
file.evt
file.undo.evt
file.trc
file.undo.trc
file.pro
file.undo.pro
```

## **trcupdate**

---

### **Related Information**

The following command: “**trcrpt**” on page 1091.

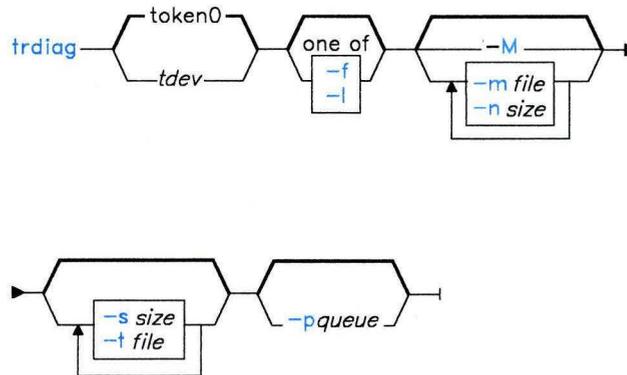
*AIX Operating System Programming Tools and Interfaces.*

# trdiag

## Purpose

Starts diagnostics on the Token-Ring Network.

## Syntax



AJ2FL129

## Description

The `trdiag` command starts the Token-Ring Diagnostics. Specify the device name of the token-ring adapter with `tdev` if you are not using `token0`.

## Flags

- `-f` Selects full error reporting. This flag cannot be used with the `-l` flag.
- `-l` Selects limited error reporting. This flag cannot be used with the `-f` flag.
- `-M` Requests logging of MAC frames. This flag uses the MAC frame log file and size specified in the `MAClog` stanza of the `/etc/trdconf` file.
- `-m file` Specifies the MAC frame log *file*. The default file is specified in the `MAClog` stanza of the `/etc/trdconf` file.
- `-n size` Specifies the *size*, in 1024 byte blocks, for the MAC frame log file. The default size is specified in the `MAClog` stanza of the `/etc/trdconf` file.

## trdiag

---

- p *queue*** Specifies the printer *queue*. If you do not specify the **-p** flag the default printer queue is used.
- s *size*** Specifies the *size*, in 1024 byte blocks, for the Token-Ring Diagnostic log file. The default is specified in the `trdlog` stanza of the `/etc/trdconf` file.
- t *file*** Specifies the *file* for the Token-Ring Diagnostic log file. The default file is specified in the `trdlog` stanza of the `/etc/trdconf` file.

## Examples

1. To start Token-Ring Diagnostics with one token-ring adapter installed:  
`trdiag`
2. To start Token-Ring Diagnostics with full error reporting enabled:  
`trdiag -f`

## Files

`/etc/trdconf`     Default configuration file.

## Related Information

"Token-Ring Diagnostics" in *Managing the AIX Operating System*.

## true

---

### Purpose

Returns an exit value of zero.

### Syntax

```
true
```

```
false
```

OL805064

### Description

The **true** command returns a zero exit value. The **false** command returns a nonzero value. These commands are usually used in input to the **sh** command.

### Example

To construct an infinite loop in a shell procedure:

```
while true
do
    date
    sleep 60
done
```

This shell procedure displays the date and time once a minute. To stop it, press **INTERRUPT (Alt-Pause)**.

### Related Information

The following command: “**sh**” on page 913.

# tsh

---

## tsh

---

### Purpose

Interprets commands in a trusted shell.

### Syntax

`^X^R1` —|

`/bin/tsh` —|

<sup>1</sup> Press Ctrl-X Ctrl-R

AJ2FL136

### Description

The AIX trusted shell (**tsh**) is a command interpreter which provides a subset of the functions of the **sh** command. The secure attention key (**Ctrl-X, Ctrl-R**) sequence (SAK) invokes the trusted shell. The trusted shell should be the login shell of the superuser. This command can also be issued by a program.

The following features are added to those of **sh** for the trusted shell (**tsh**):

- The **shell** command allows the user to return to the normal execution environment from **tsh**.
- The **logout** command allows the user to log off the system from **tsh**. This destroys any virtual terminals that the user may have opened.
- If `/bin/tsh` has the **tcb** attribute set, the user can only run trusted programs. A program must have the **tcb** attribute set to be executable by **tsh**.

The following **sh** features are not supported by **tsh**:

- **PATH** and **IFS** variable redefinition.
- Function/alias definition.

When started, this command interprets the `/etc/tsh` file. This file may contain a definition of the **PATH** variable.

---

**Japanese Language Support Information**

If Japanese Language Support is installed on your system, this command is not available.

---

**Files**

/etc/tsh\_profile      Contains initialization commands.

**Related Information**

The following commands: “**init**” on page 521 and “**shell**” on page 938

The discussion of the trusted path and **sak** in *Managing the AIX Operating System*.

## tsort

---

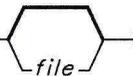
## tsort

---

### Purpose

Sorts an unordered list of ordered pairs (a topological sort).

### Syntax

`tsort` 

OL805224

### Description

The **tsort** command reads from *file* or standard input an unordered list of ordered pairs, it builds a completely ordered list, and writes it to standard output.

The input *file* should contain pairs of nonempty strings separated by blanks. Pairs of different items indicate a relative order. Pairs of identical items indicate presence, but no relative order. You can use **tsort** to sort the output of the **lorder** command.

If *file* contains an odd number of fields, **tsort** writes the error message `Odd data.`

### Example

To create a subroutine library:

```
lorder charin.o scanfld.o scan.o scanln.o \  
| tsort | xargs ar qv libsubs.a
```

This creates a subroutine library named `libsubs.a` that contains `charin.o`, `scanfld.o`, `scan.o`, and `scanln.o`. The ordering of the object modules in the library is important. The **ld** command requires each module to precede all the other modules that it calls or references. The **lorder** and **tsort** commands together add the subroutines to the library in the proper order.

---

Suppose that `scan.o` calls `scanfld.o` and `scanln.o`. `scanfld.o` also calls `charin.o`. First, the **lorder** command creates a list of pairs that shows these dependencies:

```
charin.o charin.o
scanfld.o scanfld.o
scan.o scan.o
scanln.o scanln.o
scanfld.o charin.o
scanln.o charin.o
scan.o scanfld.o
```

Next, the `|` (vertical bar) sends this list to the **tsort** command, which converts it into the ordering needed:

```
scan.o
scanfld.o
scanln.o
charin.o
```

Note that each module precedes the module it calls. `charin.o`, which does not call another module, is last.

The second `|` then sends this list to **xargs**, which constructs and runs the following **ar** command:

```
ar qv libsubs.a scan.o scanfld.o scanln.o charin.o
```

This **ar** command creates the properly ordered library.

## Related Information

The following commands: “**ar**” on page 55, “**lorder**” on page 591, and “**xargs**” on page 1232.

ttt

---

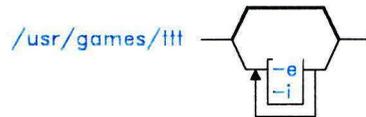
ttt

---

## Purpose

Plays tic-tac-toe.

## Syntax



OL805282

## Description

The `ttt` game plays the popular X and O game. This is a learning version, but it learns slowly. It loses nearly 80 games before completely mastering the game.

## Flags

- `-e` Increases the speed of the learning.
  - `-i` Displays the instructions prior to the start of the game.
- To quit the game, press **INTERRUPT (Alt-Pause)** or **END OF FILE (Ctrl-D)**.

## Files

/usr/games/ttt.a      Learning file.

---

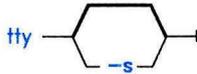
# tty

---

## Purpose

Writes to standard output the full path name of your work station.

## Syntax



OL805283

## Description

The `tty` command writes the name of your work station to standard output.

## Flag

**-s** Suppresses reporting the path name. The exit value has the following possible meanings:

- 0** Standard input is a work station.
- 1** Standard input is not a work station.
- 2** Invalid flags specified.

If your standard input is not a work station and you do not specify the `-s` flag, you get the message `not a tty`.

## Examples

- To display full path name of your work station:
- To test whether or not the standard input is a work station:

```
tty
if tty -s
then
    echo 'Enter the text to print:' >/dev/tty
fi
print
```

If the standard input is a work station, this displays the message `Enter the text to print:` as a prompt and prints the text that the user types. If the standard input is not a work station, this displays nothing. It merely prints the text read from the standard input.

The echo `... >/dev/tty` displays the prompt on the screen even if you redirect the standard output of the shell procedure. This way the prompt is never written into an output file. The special file `/dev/tty` always refers your work station, although it also has another name like `/dev/console` or `/dev/tty2`.

## turnon

---

### Purpose

Turns on execute permission for games.

### Syntax

`turnon` —|

OL805405

`turnoff` —|

OL805406

### Description

The **turnon** and **turnoff** commands are shell procedures that set the permission codes of files in the **/usr/games** directory. You must be operating with superuser authority to run this command.

The **turnon** command looks for files with permissions set to 000 and sets them to 111 (execute permission for all users).

The **turnoff** command looks for files in **/usr/games** whose permissions are set to 111 and sets these permissions to 000.

If you install any new games in the **/usr/games** directory, set their permissions to 111.

**tvi**

---

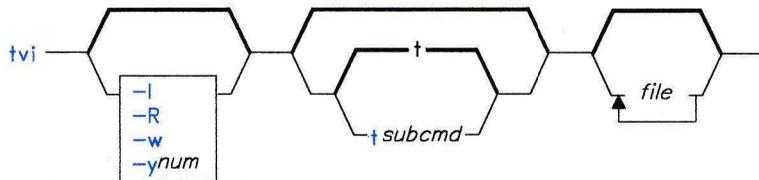
**tvi**

---

## Purpose

Acts as trusted editor for system administration.

## Syntax



OL805557

## Description

The `tvi` command provides an editor that works with a subset of the functions of the `vi` editor. It creates files only when called by the user. When auditing is enabled, and a file is edited using this command, an audit record of the type `tvi` is created.

The following `vi` features are not supported by `tvi`:

- Shell escapes
- User-defined macros
- Key mapping
- Keeping customized changes
- The `-r [file]` and `-t tag` flags
- Preserve or recover text operations
- Tags

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

## Related Information

The “`vi`, `vedit`, `view`” on page 1187 commands.

---

# ugtable

---

## Purpose

Creates, displays, and changes the Distributed Services Network Users/Groups Table.

## Syntax

`ugtable` —

OL805469

## Description

The **ugtable** command lets you build, examine, or modify the Distributed Services Network Users/Groups Table. Only members of the system group or users operating with superuser authority can use **ugtable** to change the state of the Distributed Services Network Users/Groups Table (see “su” on page 1026). Other users can use **ugtable** to browse the Network Users/Groups Table.

## Related Information

“Getting Started With Distributed Services Configuration Menus” in *Managing the AIX Operating System*.

# umask

---

## umask

---

### Purpose

Displays and sets file-creation permission code mask.

### Syntax

`umask` *nnn*

OL805286

### Description

The **umask** command sets your file-creation mask to *nnn*, three octal digits that represent the read/write/execute permissions for owner, group, and others, respectively. When you create a file, the system ANDs the complement of *nnn* to 777 for directories and 666 for files, in effect removing the corresponding permissions. (See “**chmod**” on page 160 for more information on file and directory permission codes.)

If you do not specify *nnn*, **umask** displays the current value of your file-creation permission code mask. The initial system mask (set in `/etc/profile`) is 022.

### Examples

1. To display the current file creation mask:

```
umask
```

2. To prevent other people from writing to your directories or files:

```
umask 022
```

This sets the file creation mask to 022, which takes away write permission for group members and others. Directories are created with the permission code 755. Files are created with 644.

3. To prevent other people from using your files:

```
umask 077
```

This sets the file creation mask to 077, which removes read, write, and execute permission for group members and others. Now files are created with permission code 600.

## Related Information

The following commands: “**chmod**” on page 160, “**cs**h” on page 225, and “**sh**” on page 913.

**Note:** The **cs**h command contains a built-in subcommand named **ummask**. The command and subcommand do not necessarily work the same way. For information on the subcommand, see the **cs**h command.

The **creat**, **chmod**, **mknod**, **open**, and **umask** calls in *AIX Operating System Technical Reference*.

The discussion of file permissions in *Using the AIX Operating System*.

The discussion of tailoring the user environment in *Managing the AIX Operating System*.

# umount

---

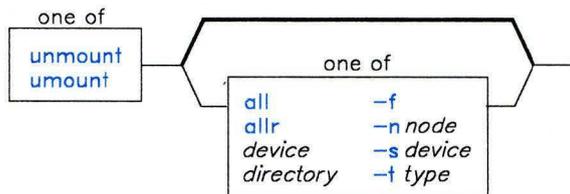
## umount, unmount

---

### Purpose

Unmounts a previously mounted file system, directory, or file.

### Syntax



OL805225

### Description

The **umount** command unmounts a previously mounted file system, directory, or file. Processing on the file system, directory, or file completes and it is unmounted. Members of the system group and users operating with superuser authority can issue any **umount** command. Other users can unmount any directory or file if they have write permission to the mounted directory or file. For local mounts, you can specify the file system, directory, or file as either the *directory* or *device* on which it is mounted.

You can also specify one of the following parameters:

- all** Unmounts all mounted file systems.
- allr** Unmounts all remote mounted file systems.

**Note:** For remote mounts, specify the directory of the file as *directory*. If you specify **allr**, **umount** unmounts all remote mounts.

### Flags

- f** Forces the unmount of one or more virtual file systems. Use in a distributed services or NFS environment to free a client when the server is down and server path names cannot be resolved.

- n node** Specifies the node which holds the mounted directory that you want to unmount. For Distributed Services, *node* can be a nickname or a node ID. For NFS, *node* can be a host name or an alias. The **umount -n node** command unmounts all remote mounts made from *node*.
- s device** Prohibits the use of the **/etc/mnttab** file if it is damaged or not writable. If you use this flag, you must specify the name of the *device* to be unmounted.
- t type** Unmounts all stanzas in **/etc/filesystems** that contain **type = type** and are mounted. (*type* is a string value, such as *remote*.)

**Note:** You cannot use the **umount** command on a device that is in use. A device is in use if any file is open for any reason or if a user's current directory is on that device.

## Examples

1. To unmount a diskette drive:

```
umount /dev/fd0
```

2. To unmount the device mounted on **/diskette0**:

```
umount /diskette0
```

3. To unmount all mounts from a remote node:

```
umount -n nodeA
```

4. To unmount files and directories of a specific type:

```
umount -t remote
```

This unmounts all files or directories that have a stanza in the **/etc/filesystems** file that contains the attribute **type = remote**.

## Files

<b>/etc/filesystems</b>	Descriptions of mountable file systems.
<b>/etc/mnttab</b>	Table of currently mounted file systems.

## Related Information

The following command: “**mount**” on page 669.

The **mount**, **umount**, **vmount**, **uvmount**, and **mntctl** system calls and the **mnttab** file in *AIX Operating System Technical Reference*.

# uname

---

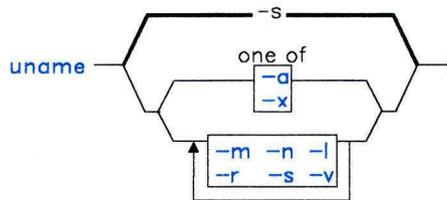
## uname

---

### Purpose

Displays the name of the current operating system.

### Syntax



OL805287

### Description

The **uname** command writes to the standard output the name of the operating system that you are using.

### Flags

- a** Displays all information specified with the **-m**, **-n**, **-r**, **-s**, and **-v** flags.
- l** Displays the LAN network number. (When Japanese Language Support is installed on your system, the **-l** flag is inactive.)
- m** Displays the type of hardware running the system.
- n** Displays the name of the node (this may be a name the system is known by to a uucp communications network).
- r** Displays the release number of the operating system.
- s** Displays the system name. (This flag is on by default.)
- S** Sets the uucp node name to the arguments parameter list.  
**Note:** You must be superuser to use this flag. No other flags are permitted.
- v** Displays the operating system version.
- x** Displays the information specified with the **-a** flag and the LAN network number. (When Japanese Language Support is installed, the **-x** flag is inactive.)

If you enter a flag that is not valid, **uname** exits with an error message, an error return status, and no output.

## Example

To display the complete system name and version banner:

```
uname -a
```

To set the node name to lance:

```
uname -S lance
```

## Related Information

The **uname** system call in *AIX Operating System Technical Reference*.

The discussion of Japanese Language Support in *Japanese Language Support User's Guide*.

# unget

---

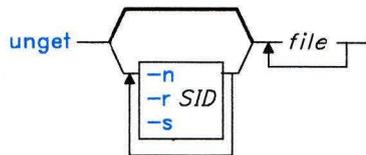
## unget

---

### Purpose

Cancels a previous **get** command.

### Syntax



OL805284

### Description

The **unget** command allows you to restore a g-file created with **get -e** before the new delta is created, and therefore discarding the changes (see “**get**” on page 477 and “**delta**” on page 310). If you specify a - (hyphen) in place of *file*, standard input is read, and each line of standard input is interpreted as the name of an SCCS file. **unget** continues to take input until it reaches an end-of-file character, which is **Ctrl-D** if entered from the keyboard.

If you specify a directory in place of *file*, **unget** performs the requested actions on all SCCS files (those files with the **s.** prefix).

### Flags

Each flag or group of flags applies independently to each named file.

- n** Prevents the automatic deletion of the g-file. This flag allows you to retain the edited version of the file without making a delta.
- rSID** Specifies the new delta that would have been created by the next use of the **delta** command. You must use this flag if you have two or more pending deltas to the file under the same login name. You can look at the p-file to see if you have more than one delta pending to a particular SID under the same login name. The *SID* specification must unambiguously specify only one SID to discard, or **unget** displays an error message and stops running.
- s** Suppresses writing the deleted SID to standard output.

## Example

To discard the changes you have made to an SCCS file after entering a `get -e`:

```
unget s.prog.c
```

## Related Information

The following commands: “**delta**” on page 310, “**get**” on page 477, and “**sact**” on page 862.

The **sccsfile** file in *AIX Operating System Technical Reference*.

The discussion of SCCS in *AIX Operating System Programming Tools and Interfaces*.

# uniq

---

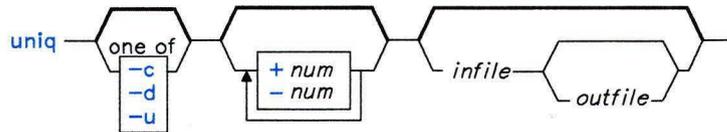
## uniq

---

### Purpose

Deletes repeated lines in a file.

### Syntax



OL805285

### Description

The **uniq** command reads standard input or *infile*, compares adjacent lines, removes the second and succeeding occurrences of a line, and writes to standard output or the specified file *outfile*. *infile* and *outfile* should always be different files. Repeated lines must be on consecutive lines in order to be found. You can arrange them with the **sort** command (see page 958) before processing.

### Flags

- c** Precedes each output line with a count of the number of times each line appears in the file. This flag supersedes **-d** and **-u**.
- d** Displays only the repeated lines.
- u** Displays only the unrepeated lines.
- num** Skips over the first *num* fields. A field is a string of nonspace, nontab characters separated by tabs and or spaces from adjacent data on the same line.
- + num** Skips over the first *num* characters. Fields specified by *num* are skipped before characters.

### Related Information

The following commands: “**comm**” on page 183 and “**sort**” on page 958.

---

# units

---

## Purpose

Converts units in one measure to equivalent units in another measure.

## Syntax

`units` —|

OL805226

## Description

The **units** command converts quantities expressed in one measurement to their equivalents in another. **units** is an interactive command. It prompts you for the unit you want to convert from and the unit you want to convert to (see “Examples” on page 1120). This command only does multiplicative scale changes. That is, it can convert from one value to another only when the conversion is done with a multiplication factor. For example, it cannot convert between degrees Fahrenheit and degrees Celsius, because 32 must be added or subtracted in the conversion.

You can specify a quantity as a multiplicative combination of units, optionally preceded by a numeric multiplier.

Indicate powers by suffixed positive integers and division by / (slash).

The **units** command recognizes **lb** as a unit of mass, but considers **pound** to be the British pound sterling. Compound names are run together (such as **lightyear**). Prefix British units differing from their American counterparts with **br** (**brgallon** for instance). The file **/usr/lib/unittab** contains a complete list of the units that the **units** command uses.

Most familiar units, abbreviations, and metric prefixes are recognized, together with the following:

<b>pi</b>	Ratio of circumference to diameter
<b>c</b>	Speed of light
<b>e</b>	Charge on an electron
<b>g</b>	Acceleration of gravity
<b>force</b>	Same as <b>g</b>
<b>mole</b>	Avogadro's number
<b>water</b>	Pressure head per unit height of water
<b>au</b>	Astronomical unit.

### Japanese Language Support Information

This command has not been modified to support Japanese characters.

---

## Examples

To start the **units** command, enter:

```
units
```

Now you can try the following examples. In these examples, the text that you enter is shown in **bold type** and the output from **units** is shown in non-bold type.

1. To display conversion factors:

```
you have: in  
you want: cm  
          * 2.540000e+00  
          / 3.937008e-01
```

The output from **units** tells you to multiply the number of inches by 2.540000e+00 to get centimeters, and to multiply the number of centimeters by 3.937008e-01 to get inches.

These numbers are in standard exponential notation, so 3.937008e-01 means  $3.937008 \times 10^{-1}$ , which is the same as 0.3937008. The second number is always the reciprocal of the first. That is,  $2.54 = 1 \div 0.3937008$ .

2. To convert a measurement to different units:

```
you have: 5 years  
you want: microsec  
          * 1.577846e+14  
          / 6.337753e-15
```

The output shows that **5 years** equals  $1.577846 \times 10^{14}$  microseconds, and that one microsecond equals  $6.337753 \times 10^{-15}$  years.

3. To give fractions in measurements:

```
you have: 1!3 mi  
you want: km  
          * 5.364480e-01  
          / 1.864114e+00
```

The ! (vertical bar) indicates division, so **1!3** means one-third. This shows that one-third mile is the same as 0.536448 kilometers.

4. To include exponents in measurements:

```
you have: 1.2-5 gal
you want: floz
          * 1.536000e-03
          / 6.510417e+02
```

The expression **1.2-5 gal** stands for  $1.2 \times 10^{-5}$ . Do *not* type an **e** before the exponent. This example shows that  $1.2 \times 10^{-5}$  (0.000012) gallons equal  $1.536 \times 10^{-3}$  (0.001536) fluid ounces.

5. To specify complex units:

```
you have: gram centimeter/second2
you want: kg-m/sec2
          * 1.000000e-05
          / 1.000000e+05
```

The units **gram centimeter/second2** mean “grams × centimeters ÷ second<sup>2</sup>.” Similarly, **kg-m/sec2** means “kilograms × meters ÷ sec<sup>2</sup>,” which is often read as “kilogram-meters per seconds squared.” Note that you can show multiplication of units with a - (hyphen) or with a blank.

6. If the units you specify after “you have” and “you want” are incompatible:

```
you have: ft
you want: lb
conformability
          3.048000e-01 m
          4.535924e-01 kg
```

The message **conformability** means that the units you specified cannot be converted. Feet measure length, and pounds measure mass, so converting from one to the other doesn’t make sense. Therefore, the **units** command displays the equivalent of each value in standard units.

In other words, this example shows that one foot equals 0.3048 meters and that one pound equals 0.4535924 kilograms. **units** shows the equivalents in meters and kilograms because the command considers these units to be “standard” measures of length and mass.

## Files

/usr/lib/unittab

# updatep

---

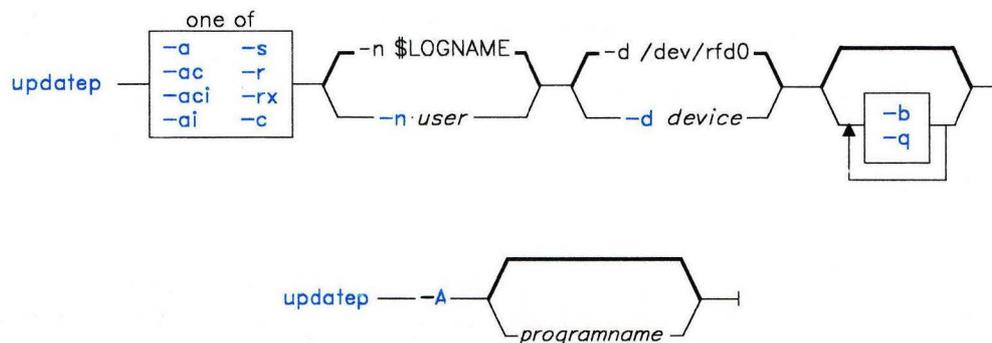
## updatep

---

### Purpose

Updates one or more programs.

### Syntax



OL805392

### Description

**Warning:** Before you apply or reject an update, restart your system and make sure no other programs are running and no other work stations are enabled.

The **updatep** command controls the update process for one or more programs. It also lets you determine the status of pending program updates and provides documentation about the updates. In addition, **updatep** can provide a list of the committed updates for each licensed program and the information changed by the update. You must be a member of the system group or operating with superuser authority to run this command.

The **updatep** command supports an apply/commit/reject philosophy. To apply one or more programs, use the **-a** or the **-ai** flags. Then use either the **-c** flag to commit the program or the **-r** flag to reject the program. Normally you do not use **-r** until you have tested the program on your system. If you specify **-ac** or **-aci**, you can apply and commit in one operation. The **-r** flag must be used separately. During an apply, **updatep** normally saves the current versions of files that are being updated. If needed, these files can be used to do a recovery or reject.

You are responsible for reserving update save space in the `/usr` file system. The **updatep** command checks to insure there is adequate save space in `/usr` before it applies an update. If there is insufficient free space, **updatep** gives you the option of either ending the command or allowing it to continue. If you end the command, you can take action to increase the free space in your `/usr` file system. If you continue, no current versions of files are saved, and **updatep** automatically commits the update, even though you may not have requested a commit originally. Normally, you should reserve 4000 blocks (2 megabytes) of free space in the `/usr` file system for updates.

You cannot use **INTERRUPT** (Alt-Pause) to stop the **updatep** command. To stop **updatep**, press **QUIT WITH DUMP** (Ctrl-V). Use only in extreme circumstances since the state of the system cannot be predicted. For example:

- The **write-verify** feature may be left on for all minidisks. See “**verify**” on page 1186.
- All terminals other than the console may be disabled. See “**pstart, penable, pshare, pdelay**” on page 791.
- Some update control files may need to be deleted.

## Flags

- a Shows authorized program analysis reports (APARs) that have been fixed for the licensed program products (LPPs) specified if none are given. This flag then shows all installed LPPs.
- a[i] Applies the updates for one or more programs. If there is a pending update for any program on the system, **updatep** does not permit an apply. You must either commit or reject all pending updates before it accepts another update apply.

The **updatep** command asks you to select the program you wish to update. After you select a program, **updatep** runs the **inudocm** command for any specific update instructions. If it finds any, it copies them into the `/usr/lpp/pgm-name/ui.vv.rr.llll` file, where *vv* is the version, *rr* the release, and *llll* the level of the program. Review instructions before continuing. To restart the update procedure and ignore the check for existing update instructions, enter `updatep -ai` or `updatep -aci`.

The **updatep** command applies the update for each program by running **inuupdt** for each name. After each update, it deletes the `/usr/lpp/pgm-name/inst_updt` directory. It then runs **inudocm** to check for any update documentation. If there is information for a manual, **updatep** copies it into the `/usr/lpp/pgm-name/me.vv.rr.lll` file and writes a message.

- A *programname* Displays a record of the applied licensed programs on your system. When used with this flag **updatep** also displays the specific areas corrected in each

## updatep

---

program. If you specify the name of the program, **updatep** displays information for only that program.

- b** Runs the **bfcreate** command to create a backup format file from the distribution media. Then tells **updatep** to use the backup file as the distribution media for the update. Use this flag to update in a code service environment.
- c** Commits a previous update apply. The **updatep** presents selection information for programs that have pending updates. You select the programs that you want to commit.  
  
Any programs that you apply as a group must be committed as a group. Management control information about the update changes to indicate that the program is committed. **updatep** deletes the directory that contains the update recovery information, **/usr/lpp/pgm-name/inst\_updt.save**.
- d device** Specifies the input device name. The default input device is **/dev/rfd0**.
- n user** Lets you specify a name in the program history file that is responsible for the program. The default is the value of the system variable **\$LOGNAME**. If you specify *user*, the first eight nonblank characters are stored in the program history file.
- q** Runs in quiet mode, suppressing most of the interactive queries.
- r** Rejects a previous update apply for one or more programs. **updatep** presents selection information for the programs that have pending updates. You select the programs to reject.  
  
Any programs that are grouped together by the system must be rejected or applied as a group. Specify **-r** without **-x** if you want automatic recovery of saved files. If you specify the **-x** flag, the management control information about the update reflects that the update is rejected, but **updatep** does not recover saved files. To recover the necessary files, look at the information in **/usr/lpp/pgm-name/inst\_updt.save**. This flag should only be used by someone very knowledgeable about the system.
- s** Writes status information about all pending program updates.
- x** Cancels the automatic recovery of saved files. (Use with **-r**.)

The **updatep** command receives the following exit codes indirectly from **update** through **inuupdt**:

- 0** Normal return, no errors indicated.
- 2** Use the **sync** command to update the super blocks, i-nodes, and delayed block I/O and then restart the AIX Operating System.
- 3** Build the kernel, then update the superblocks, i-nodes, and delayed block I/O (sync) and shut down the AIX Operating System.

- 4 Use the **cfgaply** subroutine to build the kernel. Use the **sync** command to update the super blocks, i-nodes, and delayed block I/O and then restart the AIX Operating System.
- 5 Installation cancelled without errors.
- 6 Update superblocks, i-nodes, and delayed block I/O (sync), then shut down the AIX Operating System.
- 7 Update cancelled by update procedure; recovery needed.

If it receives any other exit code, it runs the recovery function **inurecv**. If the system cannot run **updatep**, it returns an exit code of 1.

## Internal Commands

The **updatep** command uses the **inudocm** command for update documentation control. It uses the **inuupdt** command to apply an update to a single program. **inuupdt** runs a program-provided update procedure, **update**. **updatep** passes the following parameters to the **update** procedure:

- The full path name of the apply-list.
- The full path name of the device (file), where the update information is stored in **backup** format.

In addition to the commands discussed here, program-provided update procedures can use all of the internal commands discussed in “**installp**” on page 529. Since they are internal commands, they do minimum validation of input parameters. Their purpose is to provide common code for functions frequently needed by most program-provided procedures. Since these internal commands function as subcommands, they return exit values rather than issue error messages. However, messages may come from other system commands that they run. C Language programmers of update procedures that call these commands can use the `/usr/include/inu21.h` file to define the return codes for them.

### inudocm

The **inudocm** command is normally used as an internal command to get copies of specific update instructions or manual errata information that you can print out. There can be cases, however, when you would enter this command from the command line (for example, if you misplace the manual errata information that came with a previous update). You must be a member of the system group or operating with superuser authority to run this command. When auditing is on, an audit record of the type, **inudocm** is created.

The **inudocm** command has the following syntax:

```
inudocm -eu [-d device] [pgm-name] [level] [-f file]
```

where *pgm-name* specifies the name of the program being checked. It must be specified unless you use the **-f** flag. It can be a maximum of eight characters. *level* specifies the current level of *pgm-name*. This value must be identical to the level value for the last

## updatep

---

committed entry in `/usr/lpp/pgm-name/lpp.hist`. It must be specified unless you use the `-f` flag.

### Flags

- `-d device` Restores *file* from this *device*. The *device* variable must be the full path name of a device special file. The default *device* is `/dev/rfd0`. Do not specify this flag if you use the `-f` flag.
- `-e` Requests the existing update documentation information for *pgm-name* from *level*. If you select this flag, **inudocm** uses the `ar x` command to extract the archive file `/usr/sys/inst_updt/pgm-name_erata`. If this file is not present, no information is available. **inudocm** extracts any level-dependent manual errata information files if there are any more recent than the current level. Selected files are moved to `/usr/lpp/pgm-name/me.vv.rr.llll`.
- `-f file` Identifies a file that already contains the *pgm-name* and the *level*. Only **updatep** itself should use this flag.
- `-u` Requests the existing specific update instructions for *pgm-name* from *level*. If you select this flag, **inudocm** uses the `ar x` command to extract the archive file `/usr/sys/inst_updt/pgm-name_instr`. If this file is not present, no information is available. **inudocm** extracts any level-dependent specific update instruction files if there are any more recent than the current level. Selected files are moved to `/usr/lpp/pgm-name/ui.vv.rr.llll`.

The **inudocm** command returns the following exit values:

- 0** Normal return, no error occurred.
- 1** The system cannot run **inudocm**.
- 2** Specific update instruction files were requested but not found.
- 4** Manual errata information was requested, but not found.
- 6** Specific update instructions and manual errata were both requested but not found.
- 201** An invalid flag was specified, or the first argument was not `-e`, `-eu`, or `-u`.
- 202** One or more parameters were missing.
- 204** Too many parameters were entered.
- 250** The *level* parameter did not contain exactly 4 characters, or they were not numeric.
- 251** An error occurred while attempting to restore `/usr/sys/inst_updt/control`.
- 253** The directory `/usr/lpp/pgm-name` does not exist.

**inuupdt**

The **inuupdt** command provides a common interface for applying an update to a single program. Normally, **updatep** runs **inuupdt**.

The **inuupdt** command has the following format:

```
inuupdt -d device current-level new-level pgm-name
```

where *pgm-name* is the name of a program and *current-level* specifies the current maintenance level. *new-level* is the level of the update to be applied. *pgm-name* can be a maximum of eight characters and *current-level* must be identical to the level value in the */usr/lpp/pgm-name/lpp.hist* file.

The **inuupdt** command passes the following exit values to the process that called it:

0	Normal return.
2	Use the <b>sync</b> command to update the super blocks, i-nodes, and delayed block I/O, and then restart the VRM.
3	Build the kernel, then update the superblocks, i-nodes, and delayed block I/O (sync) and shut down the VRM.
4	Use the <b>cfgaply</b> subroutine to build the kernel. Use the <b>sync</b> command to update the super blocks, i-nodes, and delayed block I/O and then restart the VRM.
5	Installation cancelled without errors.
6	Update superblocks, i-nodes, and delayed block I/O (sync), then shut down the VRM.
7	Update cancelled by update procedure, recovery needed.
101-102	Places error code in the history file.
104-107	Places error code in the history file.
201-202	Places error code in the history file.
204-208	Places error code in the history file.

The **inuupdt** command returns the following exit status values:

100	Unknown return code received by <b>inuupdt</b> . It changes any unknown return code to 100 and logs it in the history file.
103	The restore of the archive file that contains the update control list, <i>/usr/lpp/pgm-name/inst_updt/arp</i> , failed.
201	An invalid flag was specified.
202	One or more parameters were missing.
203	Apply list does not exist or was not readable.
204	Too many parameters were entered.

**Flag**

**-d device** Updates the program from the specified *device*.

## updatep

---

### Files

<code>/usr/include/inu21.h</code>	Error code definitions for internal routines.
<code>/usr/lpp/pgm-name/inst-updt</code>	Temporary directory.
<code>/usr/lpp/pgm-name/inst-updt.save</code>	Directory for saved files.
<code>/usr/lpp/pgm-name/inst-updt/arp</code>	Program specific control library.
<code>/usr/lpp/pgm-name/me.vv.rr.llll</code>	Document change file.
<code>/usr/lpp/pgm-name/ui.vv.rr.llll</code>	Update instruction file.
<code>/usr/sys/inst-updt</code>	Temporary directory.
<code>/usr/sys/inst-updt/control</code>	Update control library.
<code>/usr/sys/inst-updt/inutemp.xx...x</code>	Temporary files.
<code>/usr/sys/inst-updt/pgm-name-erata</code>	Document change library.
<code>/usr/sys/inst-updt/pgm-name-instr</code>	Update instruction library.
<code>/usr/sys/inst-updt/updt-cntrl</code>	Temporary file.

### Related Information

The following commands: “**installp**” on page 529 and “**bffcreate**” on page 108.

The discussion of code service in *Managing the AIX Operating System*.

The **lpp.hist** file in *AIX Operating System Technical Reference*.

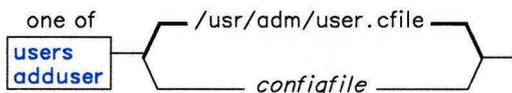
The discussion of updating programs in *AIX Operating System Programming Tools and Interfaces*.

## users, adduser

### Purpose

Adds, deletes, and changes user and group information.

### Syntax



OL805076

### Description

The **users** command lets you add, change, or delete user and group information in the **/etc/passwd**, **/etc/group**, **/etc/security/passwd**, and **/etc/security/group** files. To use the **users** command, you must be a member of the system group or have superuser authority (see “**su**” on page 1026).

The **users** command does all of its work in temporary files. When you enter the **quit** subcommand, the temporary files become the permanent files. The old versions of **/etc/passwd** and **/etc/group** are renamed **/etc/opasswd** and **/etc/ogroup**. The old versions of **/etc/security/passwd** and **/etc/security/group** are renamed **/etc/security/opasswd** and **/etc/security/ogroup**. If **users** is ended by an **INTERRUPT** (**Alt-Pause**), it removes the temporary files, and the system files remain as they were before the session. However, any directories created still exist, so you may have to remove directories after sending an **INTERRUPT**.

For configuration, **users** uses the file **/usr/adm/user.cfile**, the file specified with *configfile*, or the default parameters that follow:

Parameter	Default Value	Description
<b>udir</b>	/u/	Prefix of user home directory names.
<b>program</b>	null	The name of the user login program.
<b>siteinfo</b>	null	Any site-specific information.

Figure 12 (Part 1 of 2). Configuration File Parameters

Parameter	Default Value	Description
<b>filesize</b>	null	Size, in blocks, of the largest file that a user can make.
<b>gname</b>	staff	Name of the group to which a user is initially assigned.
<b>minid</b>	200	Minimum number that can be assigned as a user or group ID.
<b>maxid</b>	60000	Maximum number that can be assigned as a user or group ID.
<b>pfile</b>	/etc/passwd	Name of the password file.
<b>gfile</b>	/etc/group	Name of the group file.
<b>owner</b>	bin	Name of the owner of password and group files.
<b>invalid</b>	/usr/lib/sorry	Program for invalid accounts.

Figure 12 (Part 2 of 2). Configuration File Parameters

For information on how to use the **users** command, see *Managing the AIX Operating System*.

**Notes:**

1. The following files must exist on the same node:
  - /etc/passwd
  - /etc/opasswd
  - /etc/security/passwd
  - /etc/security/opasswd
  - /etc/group
  - /etc/ogroup
  - /etc/security/group
  - /etc/security/ogroup
  - /usr/adm/user.cfile
2. Each group has a limit of 500 users with eight-character IDs. Shorter IDs may allow more users per group.
3. It is possible to delete a user who still owns files or to delete a group that still has members. However, if you do this, it may cause problems later if the user name or group name is reused.

## Subcommands

add	Adds a new user or group.
change	Changes data for an existing user or group.
delete	Deletes an existing user or group.
help	Displays a summary of available commands. Entering a question mark (?) also works for help.
invalidate	Changes a user's shell to a do-nothing program.
quit	Updates files and exits.
show	Shows information about a user or group.

The initial letter of each subcommand is recognized as the subcommand name.

## Examples

The following is a sample `/usr/adm/user.cfile`:

```
pfile      /etc/passwd
gfile      /etc/group
owner      root
minid      200
maxid      1000
udir       /u/
program    /bin/sh
gname      staff
invalid    /usr/lib/sorry
```

## Files

<code>/usr/adm/user.cfile</code>	Default configuration file.
<code>/usr/adm/newuser.sys</code>	Initialization shell file for added users.
<code>/usr/adm/newuser.usr</code>	Initialization shell file for added users.
<code>/etc/passwd</code>	Password file that identifies all known users.
<code>/etc/security/passwd</code>	A file which in conjunction with <code>/etc/passwd</code> contains information about user passwords.
<code>/etc/group</code>	Group file that identifies all known groups.
<code>/etc/security/group</code>	A file which in conjunction with <code>/etc/group</code> contains information about group passwords.
<code>/etc/opasswd</code>	Saved previous version of the password file.
<code>/etc/security/opasswd</code>	Saved previous version of the security password file.
<code>/etc/ogroup</code>	Saved previous version of the group file.
<code>/etc/security/ogroup</code>	Saved previous version of the security group file.

## users

---

### Related Information

The **group** and **passwd** files in *AIX Operating System Technical Reference*.

The discussion of users and passwords in *Managing the AIX Operating System*.

---

# uucpadm

---

## Purpose

Enters basic **uucp** configuration information

## Syntax

```
uucpadm —|
```

A5AC5023

## Description

The **uucpadm** command provides interactive entry and modification of basic **uucp** configuration information for the **Devices**, **Systems**, **Permissions**, **Poll**, and **Dialcodes** files.

The **uucpadm** command uses a copy of a file to record changes. The original file remains unchanged until you enter **Ctrl-U** or **Ctrl-X** at the appropriate menu. You can use **uucpadm** repeatedly to adjust the same file.

The **uucpadm** program checks the data as it is entered. If an inconsistency in the **uucp** files is found, **uucpadm** displays a warning message.

The help routine provides instructions for each data field. Enter a ? (question mark) at any menu component to access the help routine for that field. Enter a ~ (tilde) at any menu component to edit the appropriate file for that field. The **uucpadm** command invokes the editor designated by the **EDITOR** environmental variable. If **EDITOR** is not set, **uucpadm** invokes **/usr/bin/vi**.

If your entry for the first menu item matches an existing record, **uucpadm** retrieves that record for update. The **uucpadm** program also tells you how many records have that first entry.

Initial **uucp** configuration requires that local network information be included in the **Devices**, **Systems**, and **Permissions** files, in that order. The help routine provides examples of initial entries for these files.

The **Devices** file contains profiles of all physical devices which **uucp** can use to establish a connection with a remote host. The first field in the **Devices** file is *type*, used to indicate the type of communications device.

The following keywords are recognized by *type*:

**Direct**            A direct link to another computer.  
**ACU**                An automatic call unit.  
**Built-in**          A built in or standard function, such as **TCP**.  
**System name**      A direct link to a specific computer.

The *line1* field specifies the device name associated with this entry. The *line2* field specifies the device name associated with a dialer. The *dialers* field is actually a dialer token pair field with the token optional. If the device is an automatic dialing modem, *dialers* is usually the brand name of the modem. In all cases the *dialers* field matches a record in the **Dialers** file.

A **Systems** file entry is required for each machine you communicate with. Unless *remote.unknown* is configured on your system, a **Systems** file entry is required for each machine that communicates with you. The *remote.unknown* entry allows access by unknown hosts.

The *name* field is the short node name of the remote host. The **uname -n** command can be used to obtain the *name* field. For example, if the **uname -n** command returns `lance.aus.ibm.com`, enter `lance` for *name*. The *time* field specifies when calls are permitted.

The **Permissions** file specifies options for machine access, file access, and command execution. These options take effect when a remote host logs in to your machine or you log in to a remote host.

The first prompt for an entry to the **Permissions** file is L/M. This entry must be either **LOGNAME=(login)** or **MACHINE=(short node name)**. If **LOGNAME** is entered, permissions are defined which will take effect when a remote host enters this login ID. If **MACHINE** is entered, permissions are defined which take effect when you log in to a remote host.

The **Poll** file specifies machines in your **uucp** network to be polled. Each entry contains the name of the machine to be polled and the hours the machine should be called.

The **Dialcodes** file specifies dial code abbreviations to be used in the phone number entry of the **Systems** file. Each record contains an entry for abbreviation and dial code.

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

---

## Examples

1. An entry to the **Devices** file with a direct 9600 baud connection to the lance machine on **/dev/tty2**:

```
Type = lance
line1 = tty2
line2 = -
class = 1200
dialers = direct
```

2. An entry to the **systems** file of the **lance.aus.ibm.com** system connected to an ACU device in class 2400:

```
Name = lance.aus.ibm.com
Time = Any
Type = ACU
Class = 2400
Phone = 997-7942
Login = nuucp
Password = saysme
```

3. A LOGNAME entry to the **Permissions** file:

```
L/M: LOGNAME=uucpz
Request: yes
Sendfiles: yes
Read: /
Write: NOWRITE=/etc
Callback:
Commands:
Validate: lance:backwoods
```

If the remote machine is lance or backwoods, the login ID must be uucpz. Remote hosts using this ID can Request to receive files, and your host can Sendfiles as requested. Users with this ID can read all files with *other* permissions and can write to all files, except those in */etc*, with *other* permissions.

## uucpadmin

---

4. A **MACHINE** entry to the **Permissions** file:

```
L/M: MACHINE=lance
Request: yes
Sendfiles:
Read: NOREAD=/etc
Write: NOWRITE=/etc
Callback:
Commands: ALL
Validate:
```

The machine ID is lance. Requests for file transfers can be made. The user can Read all files and can Write to all files except those in /etc. The execution of all Commands is permitted.

5. An entry to the **Poll** file consisting of the **lance.aus.ibm.com** system to be polled at 12pm, 9am, 1pm, and 6pm:

```
System = lance
Hours = 0 9 13 18
```

6. An entry to the **Dialcodes** file with LA as the abbreviation for the Los Angeles area code:

```
Abr = LA
Dialcode = 1-213-
```

## Files

/usr/adm/uucp/Devices	Information about available devices.
/usr/adm/uucp/Dialcodes	Dialing code abbreviations.
/usr/adm/uucp/Dialers	Initial handshaking on a link.
/usr/adm/uucp/Permissions	Access permission codes.
/usr/adm/uucp/Poll	Machines to be polled.
/usr/adm/uucp/Systems	Accessible remote systems.

## Related Information

The following commands: “**uucp**” on page 1144, and “**uname**” on page 1114.

---

# uucheck

---

## Purpose

Checks for files and directories required by BNU.

## Syntax

```
uucheck -v -x debug_level
```

AJ2FL107

## Description

The **uucheck** command checks for the presence of the files and directories required by the Basic Networking Utilities (BNU) facility. The command also checks for errors in the permissions file, **/usr/adm/uucp/Permissions**.

**Note:** The **uucheck** command does not check file/directory modes or some errors in the permissions file, such as duplicate login or machine names.

When BNU is installed, **uucheck** verifies that the directories, programs, and support files required to operate the networking facility are present. The command is executed automatically, as one of the first steps in the installation process, before the required BNU directories, programs, and files are actually installed.

**Note:** The **uucheck** command can be issued from the command line if the user has superuser privileges. For example, it would be useful to issue **uucheck** after making changes in part of the BNU facility such as the **Permissions** file.

## Flags

- v** Gives a detailed explanation of how the BNU programs interpret the permissions file.
- x debug\_level** Displays debugging information on the screen of the local terminal. The valid range for *debug\_level* is 0 to 9. The higher the number, the more detailed the final report.

# uucheck

---

## Files

/etc/locks/LCK*	Prevent multiple use of device.
/usr/adm/uucp/Devices	Information about available devices.
/usr/adm/uucp/Maxuuscheds	Limits scheduled jobs.
/usr/adm/uucp/Maxuuxqts	Limits remote command executions.
/usr/adm/uucp/Permissions	Access permission codes.
/usr/adm/uucp/Systems	Accessible remote systems.
/usr/spool/uucp/*	Spooling directory.
/usr/spool/uucppublic/*	Public directory.

## Related Information

The following commands: “**uucico**” on page 1139, “**uusched**” on page 1156, “**uucp**” on page 1144, “**uustat**” on page 1158, and “**uux**” on page 1166.

---

# uucico

---

## Purpose

File transport program for the BNU facility.

## Syntax

```
uucico -r role-number -x debug-level -s system-name
```

AJ2FL108

## Description

The **uucico** program transports Basic Networking Utilities (BNU) requests. The BNU commands **uucp** and **uux** both queue jobs that are transferred to the specified computer by **uucico** after the required data, work, or execute files have been created.

The **uucico** program is normally started by the scheduler, **uusched**, but it can be started manually for debugging. The BNU commands **uutry**, **Uutry**, **Nutry**, and **uukick** also start **uucico** with debugging turned on.

The **uucico** program is a BNU daemon (a program executed internally to handle file transfers and command executions). It selects the device used for the communications link, establishes the connection to the remote computer, and performs the required login sequence. The program also performs permission checks, transfers data (**D.\***) and command (**C.\***) files, logs results, and notifies specified users of transfer requests.

## Flags

- r *role\_number*** The role numbers are the number 1 for the server mode and the number 0 for client mode. The default is 0. If **uucico** is started manually, this flag should be set to 1.
- x *debug\_level*** Displays debugging information on the screen of the local terminal. The valid range for *debug\_level* is 0 to 9. The higher the number, the more detailed the final report. This flag is useful in correcting problems with the *expect-send* sequence in the **Systems** file.

## uucico

---

**-s** *system\_name*

The name of the remote system. Use only when starting **uucico** manually. The *system\_name* is supplied internally when **uucico** is started automatically. System names must contain only ASCII characters.

## Files

/etc/locks/LCK*	Prevents multiple use of device.
/usr/adm/uucp/Devices	Information about available devices.
/usr/adm/uucp/Dialcodes	Dialing code abbreviations.
/usr/adm/uucp/Dialers	Initial handshaking on a link.
/usr/adm/uucp/Maxuuscheds	Limits scheduled jobs.
/usr/adm/uucp/Maxuuxqts	Limits remote command executions.
/usr/adm/uucp/Permissions	Access permission codes.
/usr/adm/uucp/Systems	Accessible remote systems.
/usr/spool/uucp/*	Spooling directory.
/usr/spool/uucppublic/*	Public directory.

## Related Information

The following commands: “**cron**” on page 220, “**uucp**” on page 1144, “**uusched**” on page 1156, “**uustat**” on page 1158, “**uutry**, **Uutry**, **uukick**” on page 1164, and “**uux**” on page 1166.

---

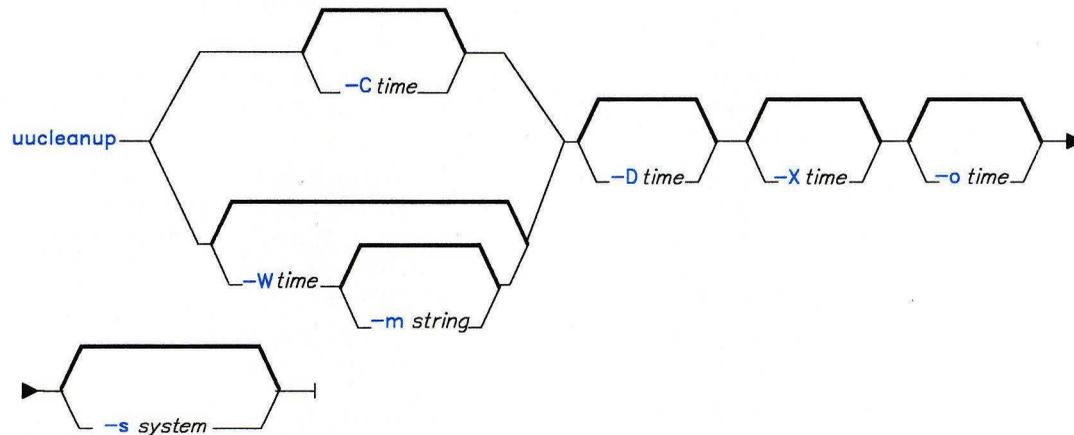
# uucleanup

---

## Purpose

Deletes selected files older than a specified number of hours from the BNU spool directory or a named directory.

## Syntax



AJ2FL109

## Description

The Basic Networking Utilities (BNU) program **uucleanup** scans the spool directory (**/usr/spool/uucp**) for old files and takes appropriate action to remove them in a useful way. Used primarily by the BNU program administrator, **uucleanup** performs the following tasks:

- Informs the requester of send/receive requests for systems that cannot be reached
- Warns users about requests that have been waiting for a given number of days; the default is 1 day
- Returns mail that cannot be delivered to the sender
- Removes all other files older than a specified number of days from the spool directory.

## uucleanup

---

The **uucleanup** program is started by the shell **uudemon.cleanup**, located in **/usr/adm/uucp**, which in turn is started by the **cron** script, located in **/usr/spool/cron/crontabs/uucp**. In general, **uucleanup** is executed automatically. You can also start the **uucleanup** program manually if you have superuser privileges.

**Note:** When BNU is installed, automatic cleanup is not enabled. Edit the file **/usr/spool/cron/crontabs/uucp** and remove the comment character “#” from the beginning of the **uudemon.cleanup** line.

### Flags

- Ctime** Removes any **C.\*** (command) files as old as, or older than, the number of days specified in *time*, and sends appropriate information to the requester. Unless specified otherwise, the default *time* is 7 days.
- Dtime** Removes any **D.\*** (data) files as old as, or older than, the number of days specified in *time*. Also attempts to deliver any remaining mail messages. The default *time* is 7 days.
- Wtime** Sends a mail message to the requester warning that **C.** files as old as, or older than, the number of days specified in *time* are still in the spool directory. The message includes the job ID and, in the case of mail, the mail message. The administrator may use the **-m** option to include a message line telling whom to call to check the problem. The default *time* is 1 day.
- Xtime** Removes any **X.\*** (execute) files as old as, or older than, the number of days specified in *time*. The default *time* is 2 days.  
**Note:** There are probably no related data files. If any related data files remain, however, they are handled by **D.\*** processing, as described above.
- mstring** Includes a specified line of text in the warning message generated by the **-Wtime** option. The default line is: See your local administrator to locate the problem.
- otime** Removes other files as old as, or older than, the number of days specified in *time*. The default *time* is 2 days.
- ssystem** Executes **uucleanup** only on the spool directory specified by *system*. The default is to clean up all BNU spool directories. System names can contain only ASCII characters.  
**Note:** Unless one of the *time* flags is set to a specific number of days, **uucleanup** uses the default *times* values.

## Files

<code>/etc/cron</code>	File that starts <b>uudemon.cleanup</b> .
<code>/usr/adm/uucp</code>	Directory with commands used internally by <b>uucleanup</b> .
<code>/usr/spool/cron/crontabs/uucp</code>	File containing <b>uudemon.cleanup</b> .
<code>/usr/spool/uucp</code>	Spooling directory.

## Related Information

The following commands: “**cron**” on page 220, “**uucp**” on page 1144, and “**uux**” on page 1166.

# uucp

---

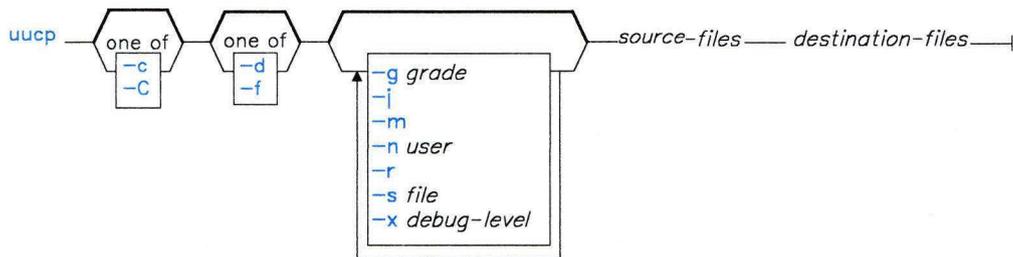
## uucp

---

### Purpose

Copies files from one AIX system to another AIX system.

### Syntax



OL805382

### Description

The Basic Networking Utilities (BNU) command **uucp** copies one or more *source* files from one AIX system to one or more *destination* files on another AIX system.

The **uucp** command accomplishes the file transfer in two steps: first, by creating a command (C.\*) file in the spooling directory on the local computer, and then by sending the request to the specified computer via the **uucico** command.

Command files include information such as the full path name of the source and destination files, the sender's login name, and so on. The full path name of a command file is a form of the following:

```
/usr/spool/uucp/system_name/C.system_nameNxxxx
```

where *N* is the grade of the request and *xxxx* is the hexadecimal sequence number used by BNU.

**Note:** If the **uucp** command is used with the **-C** flag to copy the files to the spool directory for transfer, **uucp** creates not only a command file, but also a data (D.\*) file that contains the actual source file. The full path name of a data file is a form of the following:

```
/usr/spool/uucp/system_name/D.system_namexxxx###
```

Once the command files (and data files, if necessary) are created, **uucp** then calls the **uucico** daemon, which in turn attempts to contact the remote computer to deliver the files.

**Note:** It is useful to issue the **uname** command to determine the exact name of the remote system before issuing **uucp**. The **uulog** command provides information about **uucp** activities on a system.

## Path Names Used with uucp

Path names for the source and destination of the **uucp** transfer may be one of the following:

- A full path name
- A relative path name
- A path name preceded by `~user`, where *user* is a login name on the specified system. The specified user's login directory is then considered the destination of the transfer.

If the user specifies an invalid login name, the files are transferred to the public directory, `/usr/spool/uucppublic`, which is the default.

- A path name preceded by `~/destination`, where *destination* is appended to `/usr/spool/uucppublic`.

This destination is treated as a file name unless more than one file is being transferred by this request, or the destination is a directory. To ensure that it is a directory, follow the destination name with a / (slash). For example, `~/amy/` as the destination creates the directory `/user/spool/uucppublic/amy`, if it does not already exist, and puts the requested files in that directory.

**Note:** Path names can contain only ASCII characters.

## Source and Destination File Names

- A file name can be a path name on the local system, or can have the following form:

*system\_name!path\_name*

where *system\_name* is taken from a list of system names that BNU knows about.

- The destination *system\_name* can also be a list of names, such as the following:

*system\_name!system\_name! . . . ! system\_name!path\_name*

In this case, an attempt is made to send the file via the specified route to the destination. Make sure that intermediate nodes in this route are willing to forward information (see *Managing the AIX Operating System*).

- The shell pattern-matching characters `?`, `*`, and `[ . . . ]` may be used in the path names; the appropriate system expands them.

**Note:** The shell pattern-matching characters should not be used in the path name of the destination file.

## uucp

---

- If the *destination* is a directory rather than a file, **uucp** uses the last part of the *source* name.

### Permissions

- The system administrator should restrict the access to local files by users on other systems.
- When transmitting files, **uucp** preserves execute permissions and grants read and write permissions to the owner, the group, and all others. (The **uucp** command owns the file.)
- Sending files to arbitrary *destination* path names on other systems, or getting files from arbitrary *source* path names on other systems, often fails because of security restrictions. The files specified in the path name must give read or write permission not only for the same group of users, but also for any group.
- Protected files and files in protected directories owned by the requestor can be sent by **uucp**.

**Note:** File names and system names can contain only ASCII characters.

### Flags

- c** Transfers the source files to the destination on the specified computer. The source files are not transferred via the spool directory. This saves the system from copying possibly large files to the spooling directory for transfer. (See the discussion of the **-C** flag.) This flag is on by default.
- C** Copies local files to the spool directory for transfer. Depending on the configuration of the **Poll** and **Systems** files, and on how often the **uusched** command is run, the files could be transferred immediately (on demand polling), or in the future.  
**Note:** Occasionally, there are problems in transferring a source file; for example, the remote computer may not be working, or the login attempt may fail. In such a case, the file remains in the spool directory until it is either transferred successfully or removed by the **uucleanup** command.
- d** Creates any intermediate directories needed to copy the source files to the destination. This flag is on by default.
- f** Does not create intermediate directories during the file transfer.
- ggrade** Specifies when the files are to be transmitted during a particular connection. *Grade* is a single number (0-9) or letter (A-Z, a-z); lower ASCII-sequence characters cause the files to be transmitted earlier than do higher sequence characters. The number 0 is the highest (earliest) grade; z is the lowest (latest) grade. The default is N.

- 
- j** Displays the job identification number of the transfer operation on standard output. This job ID can be used by the BNU command **uustat** to obtain the status of a information about the status of a particular job, or with **uustat -k** to terminate the transfer before it is completed.
  - m** Sends mail to the requester when the transfer to the remote system is completed. The message is sent to the requester's mailbox, **/usr/mail/user-name**. The **mail** command does not send a message for a local transfer.  
  
**Note:** The **-m** flag works only when sending files or receiving a single file. It does not work when forwarding files. Receiving multiple files specified by the shell pattern-matching characters **?**, **\***, and **[ . . . ]** does not activate the **-m** option.
  - user-name** Notifies the user specified by *user-name* on the designated system that files have been sent. The mail system does not send a message for a local transfer.  
  
**Note:** User names can contain only ASCII characters.
  - r** Prevents the starting of the file transfer program, **uucico**, even if the command was issued at a time when calls to the remote system are permitted. By default, a call to the remote system is attempted if the command is issued during a time period specified in the **Poll** and **Systems** files.
  - sfile** Reports the status of the transfer to the specified file. In this case, the *file* designation must be a full path name.
  - xdebug-level** Displays debugging information on the screen of the local system. The *debug-level* is a number between 0 and 9. The higher number gives a more detailed report.

## Examples

1. To copy one or more files locally, within the same directory:  
`uucp file1 file2`
2. To copy multiple files locally, from one directory to another directory:  
`uucp /dev/geo/project /usr/test/marg`
3. To copy file f1 from the local system to a remote system named hera:  
`uucp /u/geo/f1 hera!/u/geo/f1`
4. To copy file f2 from the remote system hera and place it in the public directory:  
`uucp hera!geo/f2 /usr/spool/uucppublic/f2`

## uucp

---

5. To place the f2 file in a directory other than the public directory:

```
uucp hera!geo/f2 /u/geo/f2
```

In this case, make sure that the geo login directory allows write permission to both “other” user and “other” group (for example, with mode 777).

## Files

/usr/spool/uucp	Spooling directory.
/usr/spool/uucppublic	Public directory.
/usr/lib/uucp	Contains <b>uucico</b> daemon.

## Related Information

The following commands: “**mail, Mail**” on page 608, “**uucleanup**” on page 1141, “**uulog**” on page 1149, “**uuname**” on page 1151, “**uusched**” on page 1156, “**uustat**” on page 1158, “**uux**” on page 1166, and “**uuxqt**” on page 1172.

The information about international character support in *Managing the AIX Operating System*.

---

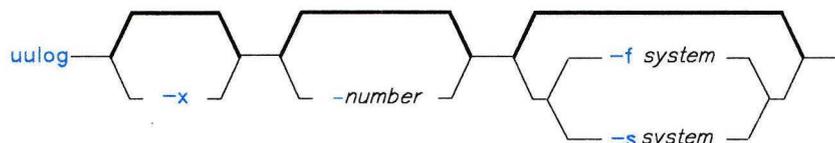
# uulog

---

## Purpose

Provides information about **uucp** and **uux** activities on a system.

## Syntax



AJ2FL111

## Description

The Basic Networking Utilities (BNU) command **uulog** displays the contents of a log file of **uucico** or **uuxqt** activities. Individual log files are created for each remote system with which the local system communicates using the **uucp**, **uuto**, or **uux** commands.

The log file of **uucico** activities is named `/usr/spool/uucp/.Log/uucico/system`. The log file of **uuxqt** activities is named `/usr/spool/uucp/.Log/uuxqt/system`.

## Flags

- f***system*      Performs a “tail -f” on the file transfer log for the specified *system*, in this case displaying the end of the log file. Use **INTERRUPT (Alt-Pause)** to leave the file and return to the prompt.
- s***system*      Prints information about copy requests involving the specified *system*.  
**Note:** System names can contain only ASCII characters.
- x**              Looks in the **uuxqt** log file for the given system.
- number**       Indicates that a **tail** command should be executed for the specified *number* of lines.

## Files

<code>/usr/bin</code>	Contains <b>uulog</b> command.
<code>/usr/spool/uucp</code>	Spooling directory.
<code>/usr/spool/uucppublic</code>	Public directory.

### Related Information

The following commands: “**tail**” on page 1044, “**uucp**” on page 1144, “**uname**” on page 1151, and “**uux**” on page 1166.

The information about international character support in *Managing the AIX Operating System*

## **uuname**

---

### **Purpose**

Provides information about other systems accessible to the local system.

### **Syntax**

`uuname -l`

AJ2FL112

### **Description**

The Basic Networking Utilities (BNU) command **uuname** displays a list of all the computers networked to the local system; the list of accessible systems is displayed on the screen of the local terminal.

In order for a local system to communicate with a remote system via BNU, the remote system must:

- have a UNIX-based operating system
- be connected to the local system.

**Note:** BNU can be used to communicate between an RT work station and a non-UNIX-based operating system, but such communications may require special hardware or software. The remote systems accessible with BNU commands are identified when the BNU programs are installed, and are listed in `/usr/adm/uucp/Systems`.

Before copying a file to another system with the **uucp** command, issue **uuname** to determine the exact name of the remote system.

### **Flags**

- l Displays the name of the local system.

## uname

---

### Examples

1. To identify the remote systems connected to the local systems:

```
uname
```

The system responds with a list like the following:

```
hera  
zeus  
merlin  
arthur
```

2. To identify the local system:

```
uname -l
```

The system responds:

```
venus
```

### Files

```
/usr/spool/uucp  
/usr/spool/uucppublic  
/usr/adm/uucp
```

```
Spooling directory.  
Public directory.  
Directory containing Systems file.
```

### Related Information

The following commands: “**uucp**” on page 1144, “**uulog**” on page 1149, and “**uux**” on page 1166.

---

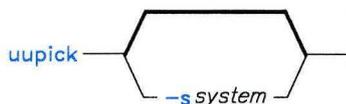
# uupick

---

## Purpose

Accepts or rejects files transmitted to a user.

## Syntax



A5AC5019

## Description

The Basic Networking Utilities (BNU) command **uupick** accepts or rejects files that the BNU command **uuto** has transmitted to a designated user **ID**.

After the files have arrived, the **rmail** command notifies the specified user. At that point, the user issues **uupick** to receive and handle the files.

Specifically, **uupick** searches the public directory on the local system for files with some form of the following name:

```
/usr/spool/uucppublic/receive/user_ID/system/file
```

For each entry (file or directory) found, **uupick** displays the following message on the screen of the local system:

```
from system: [file file-name] [dir dirname]  
?
```

It then waits for a response from standard input to determine the disposition of the file. Issuing the **uupick** command with the appropriate file-handling option completes the transfer.

# uupick

---

## File-Handling Options

After notifying the specified user that a file has been sent from *system*, **uupick** displays a question mark (?), prompting for one of the following file-handling options:

- \*** Displays all the file-handling options.
- Enter** Moves on to the next entry in the **receive** directory.
- a [dir]** Moves all **uuto** files currently in the **receive** directory into a specified directory on the local system. The default is the current working directory. Use a full or relative path name to specify *dir*.
- d** Deletes the specified file.
- m [dir]** Moves the specified file to a specified directory. If *dir* is not specified as a complete path name, a destination relative to the current directory is assumed. If no destination is given, the default is the current working directory on the local system.
- p** Displays the contents of the file on the work station screen.
- q** Stops processing and exits from the **uupick** command.
- Ctrl-D** Same as **q**.
- !cmd** Escapes to a shell to run the specified AIX command. After the command executes, returns automatically to **uupick** so the user can continue to handle the **uuto** files in the **receive** directory.

## Flags

- ssystem** Searches `/usr/spool/uucppublic/receive/user_ID/system/file` only for files sent from the specified system.

**Note:** System names can contain only ASCII characters.

## Examples

1. To receive `file1` sent with the **uuto** command from user `msg` on system `apollo`:

```
uupick
```

The system responds:

```
from system apollo: file file1  
?
```

2. Enter an asterisk (\*) to display the **uupick** file-handling options:

```
?  
*
```

The system responds:

```
usage [d] [m dir] [a dir] [p] [q] [cnt] d [!cmd] [*] [new-line]
```

Enter the appropriate option, or use the **q** option or the **Ctrl-D** sequence to exit from the **uupick** command.

## Files

`/usr/spool/uucppublic`

Public directory.

## Related Information

The following commands: “**bellmail**” on page 104, “**uuto**” on page 1162, “**uucp**” on page 1144, and “**uux**” on page 1166.

# uusched

---

## uusched

---

### Purpose

Schedules work for the BNU file transport program.

### Syntax

```
uusched -u debug_level -x debug_level
```

AJ2FL113

### Description

The **uusched** program is the Basic Networking Utilities (BNU) file-transport scheduler. It is one of the BNU daemons (a program executed internally to handle file transfers and command executions).

The **uusched** daemon schedules the transfer of files that are queued in the **/usr/spool/uucp** directory. The scheduling program first randomizes the work and then starts the **uucico** daemon with the **-s** option. This option specifies the computer for which the particular job is scheduled.

The **uusched** program itself is usually started by the shell **uudemon.hour**, which is started from **/usr/spool/cron/crontabs/uucp**, which is, in turn, started by **cron**.

### Flags

- u debug\_level** Passes as **-x debug\_level** to **uucico**. The *debug\_level* is a number from 0 to 9. Higher numbers give more detailed debugging information, which is displayed on the screen of the local system.
- x debug\_level** Outputs debugging messages from **uusched**. The *debug\_level* is a number from 0 to 9. Higher numbers give more detailed debugging information.

### Files

<b>/etc/locks/LCK*</b>	Prevents multiple use of device.
<b>/usr/adm/uucp/Devices</b>	Information about available devices.
<b>/usr/adm/uucp/Dialcodes</b>	Dialing code abbreviations.
<b>/usr/adm/uucp/Dialers</b>	Initial handshaking on a link.

/usr/adm/uucp/Permissions	Access permission codes.
/usr/adm/uucp/Systems	Accessible remote systems.
/usr/spool/cron/crontabs/uucp	Contains <b>uudemon.cleanup</b> .
/usr/spool/uucp/*	Spooling directory.
/usr/spool/uucppublic/*	Public directory.

## Related Information

The following commands: “**cron**” on page 220, “**uucico**” on page 1139, “**uucp**” on page 1144, “**uustat**” on page 1158, and “**uux**” on page 1166.

# uustat

---

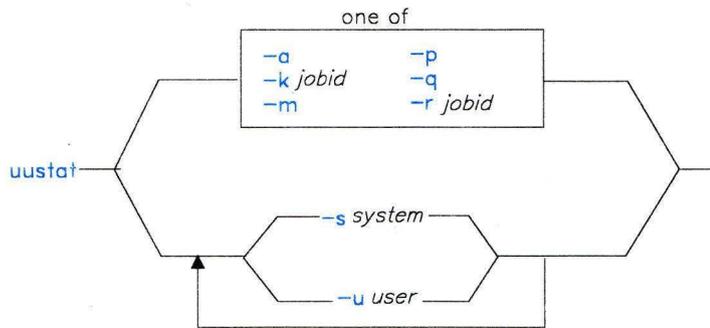
## uustat

---

### Purpose

Reports the status of and provides rudimentary job control for BNU commands.

### Syntax



AJ2FL114

### Description

The Basic Networking Utilities (BNU) command **uustat** displays status information about several types of BNU operations. It is particularly useful in monitoring transfer (copy) requests issued with the **uucp** and **uuto** commands, and requests to run an AIX command(s) on a remote system made with the **uux** command.

In addition, **uustat** also gives a user limited control over BNU jobs queued to run on remote systems. By issuing the command with the appropriate flag, a user can check the general status of BNU connections to other systems, and cancel copy requests made with **uucp** and **uuto**.

If **uustat** is issued without any flags, the command reports the status of all BNU requests issued by the current user since the last time the holding queue was cleaned up (see the description of the **-a** flag for an explanation of the BNU queues). Such status reports are displayed in the following format:

*jobid date/time status system\_name user\_ID size file*

See “Examples” on page 1160 for an explanation of this format.

**Note:** When sending files to a system that has not been contacted recently, it is a good idea to use **uustat** to see when the last access occurred, as the remote system may be down or out of service.

## Flags

The following flags are mutually exclusive; you can use only one at a time with the **uustat** command:

**-a** Displays information about all the jobs in the holding queue, regardless of the user who issued the original BNU command.

**Note:** There are two types of BNU queues.

- The current queue lists the BNU jobs either queued to run on, or currently executing on, one or more specified computers. Use the **uustat -q** command to examine this queue.
- The holding queue, accessed with the **-a** flag, lists all jobs that have not executed during a set period of time.

After the set time period has elapsed, the entries in the holding queue are deleted either manually, with the BNU command **uucleanup**, or automatically, with the file **/usr/spool/cron/crontabs/uucp** (which includes **uudemon.cleanup**), which is started by **cron**.

**-k *jobid*** Cancels (kills) the BNU process specified by the *jobid*. The person using this flag must either be the one who made the **uucp** request now being canceled, or must be operating with superuser authority.

**Note:** This flag cancels a process only when that job is still on the local computer. Once BNU has moved the job to a remote system for execution, **-k<sub>jobid</sub>** cannot be used to cancel the remote job.

**-m** Reports the status of the most recent attempt to contact the specified system with a BNU command. If the BNU request was completed, the status report is SUCCESSFUL. If the job was not completed, the status report is an error message such as LOGIN FAILED.

**-p** Runs a **ps -flp** (process status: full, long list of specified process IDs) for all PID numbers in the lock files.

**-q** Lists the jobs currently queued to run on each system; these jobs are either waiting to execute or in the process of executing. If a status file exists for the system, its date, time, and status information are reported. Once the job is finished, BNU removes that job listing from the current queue.

**Note:** In a status report, a number in parentheses next to the number of a **C.\*** (command) file or an **X.\*** (execute) file represents the age in days of the oldest **C.\***/**X.\*** file for that system. The retry field represents the number of times BNU tried and failed to execute the command because of such factors as a failed login, locked files, an unavailable device, and so on.

## uustat

---

**-r *jobid*** Marks the files in the holding queue specified by *jobid* with the current date and time. Use this flag to ensure that a cleanup operation does not delete files until the job's modification time reaches the end of the specified period.

You can use either one or both of the following flags with **uustat**:

**-ssystem** Reports the status of BNU requests for the work station specified by *system*.

**-uuser\_ID** Reports the status of BNU requests by the specified user for any work station.

**Note:** System and user names can contain only ASCII characters.

## Examples

1. To display the status of all BNU jobs in the holding queue:

```
uustat -a
```

The system responds with a display like the following:

```
heraC3113 11/06-17:47 S hera amy 289 D.venus471afd8
zeusN3130 11/06-09:14 R zeus geo 338 D.venus471bc0a
merlinC3120 11/05-16:02 S merlin amy 828 /u/amy/tt
merlinC3119 11/05-12:32 S merlin msg rmail amy
```

The first field is the job ID of the operation, which is followed by the date and time the BNU command was issued. The third field is either an S or an R, depending on whether the job is to send or request a file. The fourth field is the name of the system on which the command was entered, followed by the user ID of the person who issued the command. The sixth field is the size of the file, or, in the case of a remote execution like the last entry in the example, the name of the remote command. When the size is given, as in the first three lines of the example output, the file name is also displayed. The file name can be either the name given by the user, as in the /u/amy/tt entry, or a name that BNU assigns internally to data files associated with remote executions, such as D.venus471afd8.

2. To display the status of all jobs in the current queue:

```
uustat -q
```

The system responds:

```
merlin 3C      07/15-11:02 NO DEVICES AVAILABLE
hera    2C      07/15-10:55 SUCCESSFUL
zeus    1C (2)   07/15-10:59 CAN'T ACCESS DEVICE
```

The output tells how many C.\* (command) files are waiting for each system. The date and time refer to the current interaction with the system, followed by a report of the status of the interaction. The number in parentheses (2) in the third line of the example indicates that the C.\* file has been in the queue for two days.

3. To display all process IDs in the lock file:

```
uustat -p
LCK..tty0: 881
LCK.S.0: 879
LCK..hera: 881
F  S UID  PID PPID C  PRI NI ADDR SZ  WCHAN  STIME  TTY
101 S uucp 881 879 26 39 39 370 296 3fffe800 09:57:03 -
TIME  CMD
0:00  UUCICO -r1 -shera
101 S uuc 879 1 11 33 39 770 156 8d874 09:57:02 -
0:00  /usr/lib/uucp/uusched
```

4. To cancel a job in the current queue, first determine the job ID and then execute the **uustat -k** command:

```
uustat -a
heraC3113 11/06-17:47 S hera amy 289 D.venus471afd8
merlinC3119 11/06-17:49 S merlin geo 338 D.venus471bc0a
uustat -k heraC3113
```

5. To report the status of jobs requested by system hera:

```
uustat -s hera
heraN1bd7 07/15-12:09 S hera amy 522 /user/amy/A
heraC1bd8 07/15-12:10 S hera amy 59 D.3b2a12ce4924
heraC3119 07/15-12:11 S hera amy rmail msg
```

6. To report the status of jobs requested by user amy:

```
uustat -u amy
```

This flag displays output similar to that produced by the **-s** flag.

## Files

/etc/locks/LCK\*  
/usr/spool/uucp

Prevents multiple use of device.  
Spooling directory.

## Related Information

The following commands: “**ps**” on page 786, “**uucp**” on page 1144, “**uuto**” on page 1162, and “**uux**” on page 1166.

## uuto

---

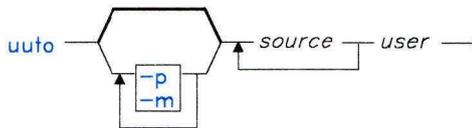
## uuto

---

### Purpose

Copies public files from one AIX system to another AIX system, with local system control of file access.

### Syntax



OL805290

### Description

The Basic Networking Utilities (BNU) command **uuto** copies one or more *source* files from one AIX system to a specified user on another AIX system. The **uuto** command calls the BNU command **uucp** for the actual file transfer, but **uuto** enables the recipient to use the **uupick** options to handle the transferred files on the local system.

The *source* entry is the name of the files on the local system, or a path name to the files on the system that runs the command. The *user* is a specific user ID. This entry has the following format:

*system!user*

where *system* is the name of a remote system connected to the local system, and *user* is the login name of the recipient of the transferred files on the specified system.

**Note:** When copying a file from one user to another user on the local system, omit the *system* entry; the destination is simply the ID of the user to whom the file is being sent. System names can contain only ASCII characters.

The **uuto** command sends files to **/usr/spool/uucppublic** on the designated *system*; this is a public directory. The command also creates an additional directory called **receive** (if it does not already exist), plus the directory */user/system*. The full path names to the copied files are therefore some form of the following:

**/usr/spool/uucppublic/receive/user/system/files**

Once the copied file is in the **receive** directory, **uuto** notifies the recipient by **rmail** that the file has arrived. The recipient then issues the **uupick** command, which searches the public directory for files sent to the specified user ID, displaying the message that file *name* has arrived from system *name* for each file it locates. The user then enters one of the **uupick** file-handling options to delete the file, move it to another directory, and so on.

## Flags

- m** Notifies the sender by **bellmail** when the copy is complete.
- p** Copies the source file to the spool directory on the local system. The source file resides in the spooling directory for a set period of time (defined in the **uusched** program) before the **uucp** command calls the **uucico** daemon, which actually transfers the copy to the public directory on the specified remote system. The default is to transfer a source file directly to the specified user.

## Files

`/usr/spool/uucppublic`          Public directory.

## Related Information

The following commands: “**bellmail**” on page 104, “**uucleanup**” on page 1141, “**uucp**” on page 1144, “**uupick**” on page 1153, “**uusched**” on page 1156, “**uustat**” on page 1158, and “**uux**” on page 1166.

## uutry

---

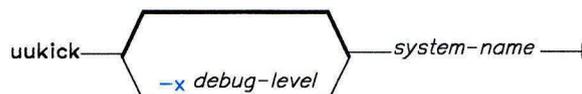
### uutry, Uutry, uukick

---

#### Purpose

Contacts a remote system with debugging turned on.

#### Syntax



AJ2FL258

#### Description

The `uutry` command contacts a specified system with debugging turned on, thus providing a means of checking call processing capabilities with debugging output. This command invokes the `uucico` program, which in turn establishes the actual connection to the remote system.

The debugging output (information about the progress of `uucico` in establishing the connection, performing the remote login, and so on) is scrolled on the screen of the local system. Once the system has finished displaying this information, use **INTERRUPT** (**Alt-Pause**) to return to the prompt.

The debugging information scrolls rapidly, so you may want to direct that output to a file by issuing `Uutry` rather than `uutry`.

The **Uutry** command (note the uppercase “U”) works almost exactly like **uutry**, with one exception. In addition to displaying the debugging output on the screen, **Uutry** also directs this information to a file named */tmp/system\_name*. Again, when the last of the output has been displayed, use **INTERRUPT** to return to the prompt.

**Note:** You can also press **INTERRUPT** while the system is scrolling the output generated by **Uutry**. This returns you to the prompt, while the **uucico** program continues to place the debugging information in */tmp/system\_name*. Use the **pg** command to examine this file.

**Note:** System names can contain only ASCII characters.

The **uukick** command also works just like the **uutry** command. The only difference between the two commands is that **uukick** takes only the **-xdebug\_level** flag. You cannot override the retry time with the **-rsystem\_name** flag.

## Flags

<b>-xdebug_level</b>	Used in debugging to override the default level of 5, and produce a detailed output of the program execution. The debugging level is a single digit between 0 and 9. Higher numbers produce more detailed debugging information, which is displayed on the screen of the local system.
<b>-r</b>	Overrides the retry time specified in <i>/usr/spool/uucp/.Status</i> .

## Files

<i>/etc/locks/LCK*</i>	Prevents multiple use of device.
<i>/tmp/system_name</i>	Temporary data file.
<i>/usr/adm/uucp/Devices</i>	Information about available devices.
<i>/usr/adm/uucp/Dialcodes</i>	Dialing code abbreviations.
<i>/usr/adm/uucp/Dialers</i>	Initial handshaking on a link.
<i>/usr/adm/uucp/Masuuscheds</i>	Limits scheduled jobs.
<i>/usr/adm/uucp/Masuuxqts</i>	Limits remote command executions.
<i>/usr/adm/uucp/Permissions</i>	Access permission codes.
<i>/usr/adm/uucp/Systems</i>	Accessible remote systems.
<i>/usr/spool/uucp/*</i>	Spooling directory.
<i>/usr/spool/uucppublic/*</i>	Public directory.

## Related Information

The following commands: “**uucico**” on page 1139, “**uucp**” on page 1144, and “**uux**” on page 1166.

## uux

---

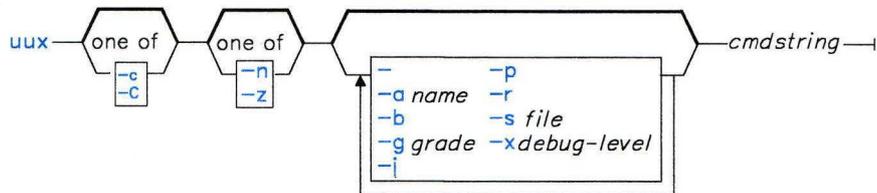
## uux

---

### Purpose

Runs a command on another AIX system.

### Syntax



AJ2FL116

### Description

The Basic Networking Utilities (BNU) command **uux** runs a specified AIX command on a specified AIX system.

The command gathers various files from the designated systems, if necessary. It then runs a specified command on a designated system. The user can direct the output from the command to a specified file on a specified system.

**Note:** For security reasons, many installations permit **uux** to run only the **rmail** command.

The **uux** command creates execute (**X.\***) files that run AIX commands on the local system. In addition, **uux** also creates both command (**C.\***) files and data (**D.\***) files. Execute files contain the command string to be executed on the designated system. Command files contain the same information as those created by the **uucp** command. Data files either contain the data for a remote command execution, or else become **X.\*** files on remote systems for remote command executions.

**Note:** The full path name of an execute file is a form of the following:

```
/usr/spool/uucp/system_name/X.system_nameNxxxx
```

After creating the files in the spooling directory, **uux** calls the **uucico** daemon, which in turn attempts to contact the designated system to deliver the files. Once the files are transferred, the **uuxqt** daemon executes the *cmdstring* on the specified system.

The *cmdstring* is made up of one or more arguments that look like an AIX command line, except that *cmdstring* may be prefixed by *system\_name*!. The default *system\_name* is the local system.

**Note:** To run commands on more than one system, type the information on separate command lines:

```
uux merlin!print /reports/memos/charles
uux zeus!print /test/examples/exampl
```

Unless the **-n** or **-z** flag is specified, **uux** notifies the user if the remote system fails to run the command. The response comes by **mail** from the other system.

## File Names, Path Names, and System Names

- When specifying the destination of the output of a command, **uux** may be entered in either one of the following formats:
  - **uux** [*options*] "*cmdstring* > *destination\_name*"
  - **uux** [*options*] *cmdstring* \{*destination\_name*\}
- Destination names may be either of the following:
  - a full path name.
  - a full path name preceded by *~user*, where *user* is a login name on the specified system. The **uux** command replaces this path name with the user's login directory.
- The shell pattern-matching characters **?**, **\***, and **[ . . . ]** can be used in the path name of a "source" file (such as files compared by the **diff** command); the appropriate system expands them. However, using the **\*** character may occasionally produce unpredictable or unanticipated results.
 

**Note:** Shell pattern-matching characters should not be used in the destination path name.
- Place either two backslashes (**\ . . . \**) or a pair of quotation marks (**" . . . "**) around pattern-matching characters in a path name so the local shell cannot interpret them before **uux** sends the command to a designated system.
- If using the special shell characters **>** (greater than), **<** (less than), **;** (semicolon), or **|** (vertical bar) in a path name, place either **\ . . . \** or **" . . . "** around the individual character or around the entire command string.
- Do not use the shell redirection characters **<<** or **>>** in a path name.
- The **uux** command attempts to move all files specified on the command line to the designated system. Enclose the names of all output files in parentheses so that **uux** does not try to transfer them.
- When specifying a *system\_name*, always place it before the *cmdstring* in the entry.

**Note:** System names can contain only ASCII characters.

- The exclamation point preceding the name of the local system in a command is optional. If you choose to include the ! to run a command on the local system using files from two different remote systems, use ! instead of *system!* to represent the local system, and add *system!* as the first entry in any path name on the remote systems.
- The exclamation point representing a remote system in BNU syntax has a different meaning in C shells (**cs**h). When running **uux** in a C shell, place a backslash (\) before the exclamation point in a system name.
- If the command being executed requests two files stored on the same system, or two files with the same name that are stored on separate systems, the command will execute, but will not produce the desired results.

The following two commands *will* execute:

```
uux "hera!/bin/diff /usr/amy/out1 hera!/u/amy/out > ~uucp/DF"
uux "hera!/bin/diff hera!/usr/amy/out1 venus!/u/amy/out > ~uucp/DF"
```

**Note:** The notation ~uucp is the shorthand way of specifying the public spooling directory **/usr/spool/uucppublic**.

In the first command, `diff` is on system `hera`, the first source file is on the local system, the second source file (with a different name) is on system `hera`, and the output is directed to the file `DF` in the public directory on the local system. In the second command, `diff` is again on `hera`, the first file is also on `hera`, the second file (with a different name) is on `venus`, and the output is again directed to `DF` in the ~uucp directory.

The following command will not execute properly:

```
uux "hera!/bin/diff venus!/u/amy/out merlin!/u/amy/out > ~uucp/DF"
```

This command will not execute because, although the files are on two different systems, they still have the same file name.

## Flags

- Makes the standard input to **uux** the standard input to the *cmdstring*.
- aname* Replaces the user ID of the person issuing the command with user ID specified with *name*.
- b** Returns standard input to the command if the exit status is not zero.

- 
- c** Transfers the source files to the destination on the specified system. The source files are not copied into the spool directory for transfer. (See the discussion of the **-C** flag.) This flag is on by default.
- C** Transfers the source files to the spool directory. After a set period of time (specified in the **uusched** program), the **uucico** daemon attempts to transfer the files to the destination on the specified computer.
- Note:** Occasionally, there are problems in transferring a source file; for example, the remote computer may not be working, or the login attempt may fail. In such cases, the file remains in the spool directory until it is either transferred successfully or removed by the **uucleanup** command.
- ggrade** Specifies when the files are to be transmitted during a particular connection. *Grade* is a single number (0-9) or letter (A-Z, a-z); lower ASCII-sequence characters cause the files to be transmitted earlier than do higher sequence characters. The number 0 is the highest (earliest) grade; z is the lowest (latest). The default is **N**.
- j** Displays the job identification number of the process that is running the command on the specified system. Use this job ID with the **BNU** command **uustat** to check the status of the command, or with **uustat -k** to terminate the process.
- n** Prevents user notification by **mail** of the success or failure of a command. The default is to notify the user if the command fails.
- p** Uses the standard input to **uux** as the standard input to *cmdstring*. **A -** (minus) has the same effect.
- r** Prevents the starting of the spooling program that transfers files between systems. The default is to start the spooling program.
- sfile** Reports the status of the transfer in a file specified by *file* on the designated system.
- Note:** File names can contain only ASCII characters.
- xdebug\_level** Displays debugging information on the screen of the local system. The *debug\_level* is a number between 0 and 9. The higher number gives a more detailed report.
- z** Notifies the user if the command completes successfully. This flag is the opposite of the system default, which is to notify the user of failure.

## Examples

1. To get the *jobid* of a job and then compare a file on the local system ZEUS with a file on a remote system when the **diff** command is stored on the local system, use either of the following formats:

```
uux -j "/bin/diff /usr/amy/f1 hera!/u/amy/f2 > ~uucp/f1.diff"
```

or

```
uux -j /bin/diff /usr/amy/f1 hera!/u/amy/f2 \{~uucp/f1/diff\}
```

This command gets the file `/usr/amy/f1` from the remote system `hera`, compares it to the file `/u/amy/f2` on the local system (ZEUS), and places the output of the command in the local public directory in a file named `f1.diff`. (The full path name of this file is `/usr/spool/uucppublic/f1.diff`.) Using the `-j` option produces the output `zeusN52d9`.

**Note:** As shown in the example, the destination name must be entered either preceded by a `>` with the whole command string enclosed in `" . . . "`, or entered enclosed in braces and backslashes, as `\{ . . . \}`.

2. To compare files that are located on two different remote systems, `hera` and `venus`, using the **diff** command on the local system:

```
uux "!/bin/diff hera!/usr/amy/f1 venus!/u/amy/f2 > !f1.diff"
```

This command gets the `/usr/amy/f1` file from the system `hera` and the `/u/amy/f2` file from `venus`, runs a **diff** command on the two files, and places the results in the file `f1.diff`, located in the current working directory on the local system.

- This output file must be write-enabled. If you are uncertain about the permission status of a specific target output file, direct the results to the public directory, as in the first example.
- The exclamation points representing the local system are optional.
- Both of the examples above use a `>` symbol preceding the name of the output file. When using the special shell characters `>`, `<`, `;`, or `|`, either put the entire *cmdstring* in quotation marks, or put the special characters as individual arguments in quotation marks.

3. To specify an output file on a different remote system:

```
uux hera!uucp venus!/u/amy/f1 \{merlin!/u/geo/test\}
```

This command runs **uucp** on system `hera`. The **uucp** command then sends the file `/u/amy/f1`, stored on system `venus`, to user `geo` on system `merlin` as `test`.

4. To get selected fields from a file on system `hera` and place them in a file on the local system:

```
uux "cut -f1 -d: hera\!/etc/passwd > ~uucp/passw.cut"
```

This command runs `cut` on the local system, gets the first field from each line of the password file on system `hera`, and places the output in the file `passw.cut` in the public directory on the local system.

**Note:** In this example, `uux` is running in a `c` shell, so a `\` (backslash) must precede the exclamation point in the name of the remote system.

## Files

`/usr/spool/uucp`  
`/usr/lib/uucp`

Spooling directory.  
Contains the `uucico` daemon.

## Related Information

The following commands: “`mail`, `Mail`” on page 608, “`uucico`” on page 1139, “`uucp`” on page 1144, “`uustat`” on page 1158, and “`uuxqt`” on page 1172.

## uuxqt

---

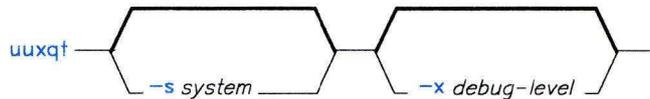
## uuxqt

---

### Purpose

Executes remote command requests.

### Syntax



AJ2FL117

### Description

The Basic Networking Utilities (BNU) program **uuxqt** executes specified commands on designated remote systems.

Once **uux** is entered by a user, the program creates the necessary command (**C.\***), data (**D.**), and execute (**X.\***) files and places them in the spooling directory on the designated system. The **uux** command then calls the **uucico** daemon, which in turn attempts to contact the designated system to deliver the files. When the files have been transferred, **uuxqt** executes the commands on the designated system.

The **uuxqt** program searches the spool directories on the designated system for **X.\*** files whose names indicate that they have been sent from another system. The command checks each execute file to make sure that:

- All the required data (**D.\***) files are available and accessible
- File commands are permitted for the requesting system.

**Note:** BNU uses the **Permissions** file to validate file accessibility and command execution permission.

The **uuxqt** program, one of the Basic Networking Utilities (BNU) daemons (a program executed to handle file transfers and command executions), can be executed manually by an individual with superuser privileges. This daemon is executed automatically by the **uudemon.hour** shell script, which is started periodically by **cron**.

---

## Flags

- ssystem** The name of the remote system. Use only when starting **uuxqt** manually. The *system* name is supplied internally when **uuxqt** is started automatically.
- Note:** System names can contain only ASCII characters.
- xdebug\_level** Displays debugging information on the screen of the local system. The *debug\_level* is a single digit between 0 and 9. The higher the number, the more detailed the debugging information.

## Files

/etc/locks/LCK*	Prevents multiple use of device.
/usr/adm/uucp/Maxuuxqts	Limits remote command executions.
/usr/adm/uucp/Permissions	Access permission codes.
/usr/spool/uucp/*	Spooling directory.

## Related Information

The following commands: “**uucico**” on page 1139, “**uucp**” on page 1144, “**uustat**” on page 1158, and “**uux**” on page 1166.



## **Related Information**

The following commands: “**fsck**, **dfsck**” on page 445, “**umount**, **unmount**” on page 1112, “**varyon**” on page 1180, and “**vrmdir**” on page 1206.

The discussion of external drives in *Managing the AIX Operating System*.

The **system** and **filesystems** files in *AIX Operating System Technical Reference*.

# varyon

---

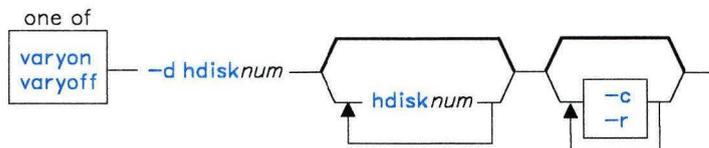
## varyon

---

### Purpose

Makes an external disk drive and any minidisks or file systems defined on it available for use.

### Syntax



OL805455

### Description

The **varyon** command adds an external disk drive and any minidisks or file systems residing on the disk to the existing AIX Operating System hardware and minidisk configurations. **varyon** runs the **fsck** command to perform file system consistency checks and mounts the file system if the attributes **vcheck = true** and **vmount = true** are present in the corresponding stanza of the **/etc/filesystems** file. By default, **varyon** sends only generalized completion messages to standard output, routing more detailed error messages to the file **/etc/varyon.out**.

### Flags

- c** Specifies that **varyon** not perform file consistency checking.
- d hdisknum . . .** Specifies the system device name for an external disk drive, where *num* is an integer from 0 to 33, inclusive. The **varyon** command uses **vrconfig** to configure minidisks residing on the drive. It uses **fsck** to check each file system whose stanza in the **/etc/filesystems** file contains the attribute **vcheck = true**. It mounts each file system that has the attribute **vmount = true**.
- q** Operates in quiet mode; does not prompt the user about minidisk names or rename the minidisks. If **/etc/system** and **/etc/filesystems** contain information about minidisks on the external disk drive, this option adds the minidisks to the system configuration using the existing minidisk names and mount directory.

---

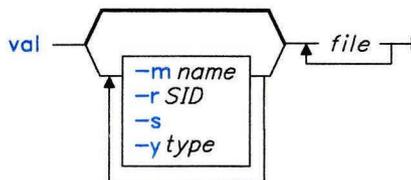
# val

---

## Purpose

Validates Source Code Control System (SCCS) files.

## Syntax



OL805292

## Description

The **val** command reads *files* and determines if the specified *file* is an **Source Code Control System (SCCS)** file meeting the characteristics specified by the flags. If you specify a - (minus) for *file*, **val** reads standard input and interprets each line of standard input as **val** flags and the name of an SCCS file. **val** continues to take input until it reaches an end-of-file character (**Ctrl-D**).

The **val** command displays error messages to standard output for each file processed. **val** also returns a single 8-bit code upon exit. The 8-bit code indicates possible mismatches or errors. It is interpreted as a bit string in which set bits (from left to right) are interpreted as follows:

- bit 0 = missing file parameter
- bit 1 = unknown or duplicate flag
- bit 2 = damaged SCCS file
- bit 3 = cannot open file or file not SCCS
- bit 4 = SID is invalid
- bit 5 = SID does not exist
- bit 6 = %Y%, -y mismatch
- bit 7 = %M%, -m mismatch

When **val** processes two or more files on a given command line or multiple command lines (when reading the standard input), a code is returned that is a logical OR of the codes generated for each command line and file processed. **val** can process up to 50 files on a single command line. Any number above 50 produces a dump.

## Flags

Each flag or group of flags applies independently to each named file. The flags can appear in any order.

- mname** Compares the value *name* with the SCCS %M% identification keyword in *file*. See “Identification Keywords” on page 480 for more information on the %M% keyword.
- rSID** Specifies the *SID* of the *file* to be validated. The *SID* must be valid and unambiguous.
- s** Suppresses the error message normally written to standard output.
- ytype** Specifies a *type* to compare with the SCCS %Y% identification keyword in *file*. See “Identification Keywords” on page 480 for more information on the %Y% keyword.

## Related Information

The following commands: “**admin**” on page 41, “**delta**” on page 310, “**get**” on page 477, and “**prs**” on page 781.

The **sccsfile** file in *AIX Operating System Technical Reference*.

The discussion of SCCS in *AIX Operating System Programming Tools and Interfaces*.

---

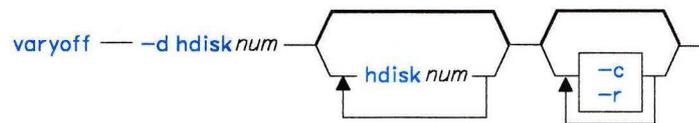
# varyoff

---

## Purpose

Removes an external disk drive from the operating system configuration.

## Syntax



AJ2FL119

## Description

The **varyoff** command removes an external disk drive and any minidisks residing on the disk from the existing AIX Operating System hardware and minidisk configurations. File systems residing on minidisks are unmounted and file system consistency checks are performed.

## Flags

- c** Does not perform file consistency checking.
- d hdisknum . . .** Specifies the system device name for an external disk drive, where *num* is an integer from 0 to 33, inclusive. The corresponding drive and the minidisks residing on that drive are removed from the AIX Operating System. **umount** and **fsck** are performed on each file system defined on the disk.
- r** Removes information about all minidisks on the external disk drive from the files **/etc/system** and **/etc/filesystems**.

## varyoff

---

### Examples

1. To remove a disk from the AIX Operating System using its name:

```
varyoff -d hdisk7
```

This command **unmounts** and performs **fsck** functions on each file system defined on the disk drive named hdisk7. hdisk7 and all of its minidisks are removed from the operating system.

2. To remove more than one disk:

```
varyoff -d hdisk9 hdisk7 hdisk12
```

This command removes disk drives hdisk9, hdisk7, and hdisk12 from the operating system.

3. To remove information about all minidisks on an external disk drive from the system configuration files:

```
varyoff -r -d hdisk10
```

This command removes information about all minidisks on the external disk drive hdisk10 from the files **/etc/system** and **/etc/filesystems**.

4. To avoid **fsck** functions when removing a disk:

```
varyoff -c -d hdisk7
```

### Files

<code>/etc/filesystems</code>	Descriptions of mountable file systems.
<code>/etc/system</code>	Default system file.
<code>/etc/system.vary.bk</code>	System backup of <b>/etc/system</b> .
<code>/etc/varyoff.out</code>	Default message output file.

## Examples

1. To configure an entire external disk drive:

```
varyon -d hdisk7
```

This command configures a disk drive, `hdisk7`, into the operating system, configures any minidisks defined on the disk, and performs **fsck** and **mount** functions on file systems as specified by the `/etc/filesystems` file.

2. To configure more than one external disk drive:

```
varyon -d hdisk9 hdisk7 hdisk12
```

This command makes disk drives `hdisk9`, `hdisk7`, and `hdisk12` available for use.

3. To configure an external drive without performing **fsck** functions:

```
varyon -c -d hdisk7 hdisk8 hdisk11
```

4. To make the external disk drive available without prompting the user, even if minidisks are already defined in the system configuration files:

```
varyon -q -d hdisk3
```

This command adds the minidisks to the system without prompting the user if `/etc/system` and `/etc/filesystems` contain information about any of the minidisks on the drive. If information exists the system uses the existing minidisk names and mount directory.

## Files

<code>/etc/filesystems</code>	Descriptions of mountable file systems
<code>/etc/system</code>	Default system file
<code>/etc/system.vary.bk</code>	System backup of <code>/etc/system</code>
<code>/etc/varyon.out</code>	Default message output file

## Related Information

The following commands: “**fsck**, **dfsck**” on page 445, “**mount**” on page 669, “**varyoff**” on page 1177 and “**vrconfig**” on page 1206.

The discussion of external drives in *Managing the AIX Operating System*.

The **system** and **filesystems** files in *AIX Operating System Technical Reference*.

vc

---

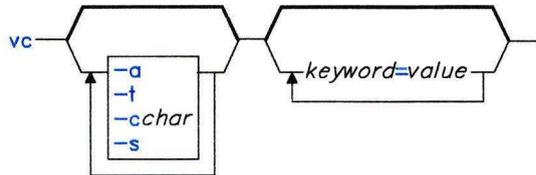
vc

---

## Purpose

Substitutes assigned values in place of keywords.

## Syntax



OL805293

## Description

The `vc` command is used to control writing different versions of a file to standard output. However, since Source Code Control System commands (“**admin**” on page 41, “**get**” on page 477, and “**delta**” on page 310) provide more efficient and complete control, they should be used instead of `vc`.

The `vc` command copies lines from standard input to standard output. The flags and *keywords* on the command line and control statements in the input modify the resulting output. `vc` replaces user declared *keywords* with their *value* assigned on the command line. These *keywords* can be replaced both in text and in control statements.

## Control Statements

A control statement is a single line beginning with a control character (the default control character is a `:` (colon)). They provide conditional processing of the input. The allowable types of control statements are:

**:if** *condition*

*text*

**:end**

Writes all the lines between the **:if** statement and the matching **:end** to standard output only if *condition* is true. You can nest **:if:~end** statements, but once a condition is false, all remaining nested **:if:~end** statements are ignored. See “Condition Syntax” on page 1183 for the syntax of conditions and allowable operators.

---

**:dcl** *keyword* [, *keyword* . . . ]  
 Declares *keywords*. All *keywords* must be declared.

**:asg** *keyword* = *value*  
 Assigns *value* to *keyword*. An **:asg** statement takes precedence over keyword assignment on the **vc** command line. A later **:asg** statement overrides all earlier assignments of the associated *keyword*. *keywords* declared, but not assigned *values* have null values.

**::** *text*  
 Removes the two leading control characters and replaces *keywords* with their *values*, and then copies the line to the standard output.

**:on**  
**:off**  
 Turns on or off *keyword* replacement on all lines.

**:ctl** *char*  
 Changes the control character to *char*.

**:msg** *message*  
 Writes a message to standard error output in the form:  
 Message(*n*): *message*  
 where *n* is number of the input line on which the message appeared.

**:err** *message*  
 Writes an error message to standard error in the form:  
 ERROR: *message*  
 ERROR: err statement on line *n* (vc15)  
**vc** stops processing and returns an exit *value* of 1.

## Condition Syntax

Item	Statements Allowed
condition	::=or statement ::=not or statement
or statement	::=and statement ::=and statement   or statement
and statement	::=expression ::=expression & and statement
expression	::=( or statement ) ::=value operator value
operator	::== or != or < or >
value	::=ASCII string

Item	Statements Allowed
	::=numeric string

The available condition operators and their meanings are as follows:

=	equal
!=	not equal
&	and
	or
>	greater than
<	less than
()	used for logical groupings
<b>not</b>	may only occur immediately after the <i>if</i> , and when present, inverts the value of the entire condition.

The > and < (greater than and less than symbols) operate only on unsigned integer values; for example: 012 > 12 is false. All other operators take strings as modifiers; for example 012 != 12 is true. The precedence of the operators, from highest to lowest precedence, is as follows:

```

= != > < all of equal precedence
&
|

```

Parentheses can be used, of course, to alter the order of precedence.

Values must be separated from operators or parentheses by at least one blank or tab.

## Keyword Replacement

A *keyword* must begin and end with the same control character used in control statements. A *keyword* may be up to nine alphanumeric characters, where the first character must be alphabetic. Keyword *values* can be any ASCII string. A numeric keyword *value* is an unsigned string of digits. *values* cannot contain tabs or spaces.

Examples of *keyword*=*value* assignments are:

```

numlines=4
prog=acctg
pass4=yes

```

The **vc** command removes all control characters and *keywords* from input text lines marked with two control characters as it writes the text to standard output. To prevent a control character from being interpreted, precede it with a backslash, as in the following example:

```

::the :prog: program includes several of the following\:
```

The keyword **prog**: is replaced by its *value*, but the \: is passed to standard output as : (colon).

---

Input lines beginning with a \ (backslash) followed by a control character are not control lines, and are copied to standard output without the backslash. **vc** writes lines beginning with a backslash and no following control character without any changes (including the initial backslash).

## Flags

- a Replaces *keywords* surrounded by control characters with their assigned *value* in all text lines (not just those beginning with two control characters).
- c*char* Uses *char* as the control character.

---

### Japanese Language Support Information

*char* must be an ASCII character.

---

- s Does not display the warning messages normally displayed to standard error.
- t Ignores all characters from the beginning of a line up to and including the first tab character for detecting a control statement. If **vc** finds a control character, it ignores all characters up to and including the tab.

## Related Information

The following commands: “**admin**” on page 41, “**delta**” on page 310, and “**get**” on page 477.

The discussion of Japanese Language Support in *Japanese Language Support User's Guide*.

# verify

---

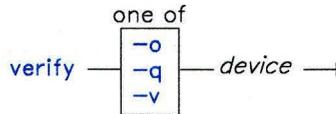
## verify

---

### Purpose

Turns write verification on or off for a particular minidisk.

### Syntax



OL805446

### Description

The **verify** command turns write verification on or off for the specified minidisk *device*. The name must be a stanza name in the */etc/system* file.

When write verification is on, the system checks each write operation to the minidisk by comparing the data written to disk with the data in the write buffer. If it detects an uncorrectable error, then it passes an error code back from the write operation.

At system startup, write verification is turned off.

### Flags

- o Turns write verification off.
- q Tells you whether write verification is set on or off.
- v Turns write verification on.

### Files

*/dev/config*  
*/etc/system*

### Related Information

The **mdverify** subroutine in *AIX Operating System Technical Reference*.

---

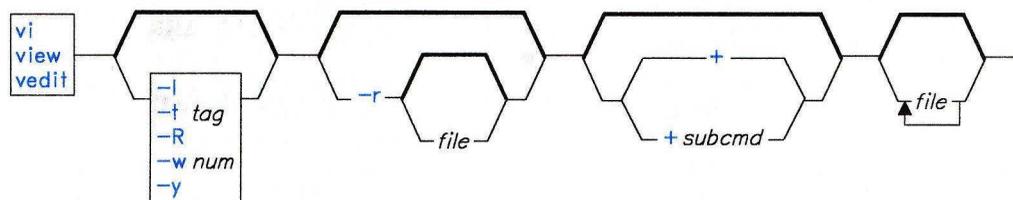
## vi, vedit, view

---

### Purpose

Edits files with a full screen display.

### Syntax



OL805424

### Description

The **vi** command starts a display editor based on an underlying line editor (**ex**). The **view** command is a read-only version of **vi**. In it, the **readonly** option is set to protect files during browsing. The **vedit** command is a version of **vi** intended for beginners. In it, the **report** option is set to 1, and the **showmode** and **novice** options are set. Since **novice** is set, it is a line editor. For more information on these options, see “Setting Options” on page 1189.

The *file* parameter specifies the file or files to be edited. If you supply more than one *file* on the command line, **vi** edits each file in the specified order.

When you use **vi**, changes made to a file are reflected automatically in your display. The position of the cursor on the display indicates its position within the file. The subcommands affect the file at the cursor position.

The following list provides the maximum limits of the **vi** editor. These counts assume single-byte characters.

- 2048 characters per line
- 256 characters per global command list
- 128 characters in the previous inserted and deleted text
- 128 characters in a shell escape command
- 128 characters in a string-valued option
- 30 characters in a tag name
- 1,048,560 lines of 2048 characters per line silently enforced
- 128 map macros with 2048 characters total.

## Editing States

The **vi** editor has the following operational states:

command state	This is the initial state. Any subcommand can be entered (except commands that can only be used in the text input state). When subcommands and other states end, they return to this state. Pressing <b>Esc</b> cancels a partial command.
text input state	You use <b>vi</b> in this state when you add text. Entered this state with the <b>a</b> , <b>A</b> , <b>i</b> , <b>I</b> , <b>o</b> , <b>O</b> , <b>c</b> , <b>C</b> , <b>s</b> , <b>S</b> , and <b>R</b> subcommands. After entering one of these commands, you can enter text into the editing buffer at the current cursor position. To return to the command state, press <b>Esc</b> for normal exit or press <b>INTERRUPT (Alt-Pause)</b> to end abnormally.
last line state	Some subcommands (subcommands with the prefix <b>:</b> , <b>/</b> , <b>?</b> , <b>!obj</b> , or <b>!!</b> ) read input on a line displayed at the bottom of the screen. When you enter the initial character, <b>vi</b> places the cursor at the bottom of the screen, where you enter the remaining characters of the command. Press the <b>Enter</b> key to run the subcommand and <b>INTERRUPT (Alt-Pause)</b> to cancel it. When <b>!!</b> is used, the cursor moves only after both <b>!</b> s are entered. When you use <b>:</b> (colon) to enter the last line state, special meaning is given to the following characters when they are used before commands which specify counts.
	<b>%</b> All lines regardless of cursor position
	<b>\$</b> Last line
	<b>.</b> Current line

## Character Sets with vi

The collation table defines the alphanumeric set used by your system. This table affects the performance of **vi** macros and subcommands. If you intend to use non-ASCII extended characters with **vi** macros, it may be necessary to revise this table using the **ctab** command.

The **vi** editor uses the collation table to distinguish between a *small word* and a *big word*. A small word is bounded by alphabetic characters or numbers that are not defined in the collation table. For example, **i sn't** is two small words. The **'** (apostrophe) is not a number or an alphabetic character, and it bounds both the small word **t** and the small word **i sn**. A big word is bounded by blanks, tabs, and new-line indicators. For example, **stop** is a big word. For more information see “**ctab**” on page 257 in this book. For more information on extended characters, international characters, and collation tables see *Managing the AIX Operating System*.

## Setting Options

The **vi** editor allows you to customize options so that you can use the editor for a specific task. Use the **set** command to set or change an option. You set some options to a string or a number value, other options you simply turn on or off. To change an option set to a value, enter a command in the form **:set option = value**. To toggle an option that can be set to on or off, enter a line of the form **:set option** to set it on or **:set no option** to set it off. To view the current setting of all the options, enter **:set all** while in the **vi** command state.

You can abbreviate options with the **set** command. The following table lists commonly used options, abbreviations, and descriptions:

Option	Abbreviation	Description
<b>autoindent</b>	<b>ai</b>	Indents automatically in text input mode to the indentation on the previous line by using the spacing between tab stops specified by the <b>shiftwidth</b> option. The default is <b>noai</b> . To back the cursor up to the previous tab stop, type <b>Ctrl-D</b> . This option is not in effect for global commands.
<b>autoprint</b>	<b>ap</b>	Prints the current line after any command that changes the editing buffer. The default is <b>ap</b> . This option applies only to the last command in a sequence of commands on a single line, and is not in effect for global commands.
<b>autowrite</b>	<b>aw</b>	Writes the editing buffer to the file automatically before the <b>:n</b> , <b>:ta</b> , <b>Ctrl-A</b> , and <b>!</b> subcommands if the editing buffer changed since the last <b>write</b> command. The default is <b>noaw</b> .
<b>beautifying text</b>	<b>bf</b>	Prevents the user from entering control characters (except for tab, new-line, and scans form feed) in the editing buffer during text entry. The default is <b>nobf</b> . This option applies to command input.
<b>closepunct</b>	<b>cp =</b>	Handles a list of closing punctuation specially when wrapping text (see <b>wraptyp</b> ). Precedes multi-character punctuation with the number of characters. Example: <b>cp=3 . . ; ) }</b> . The <b>vi</b> command does not split closing punctuation when wrapping. To set the default value, run kanji-compatible <b>vi</b> and enter <b>:set cp</b> .
<b>directory</b>	<b>dir =</b>	Displays the directory that contains the editing buffer. The default is <b>dir = /tmp</b> .

---

Option	Abbreviation	Description
<b>edcompatible</b>	<b>ed</b>	Retains global ( <b>g</b> ) and confirm ( <b>c</b> ) subcommand suffixes during multiple substitutions and causes the read ( <b>r</b> ) suffix to work like the <b>r</b> subcommand. The default is <b>noed</b> .
<b>hardtabs</b>	<b>ht =</b>	Tells <b>vi</b> the distance between the hardware tab stops on your display. The default is <b>ht = 8</b> .
<b>ignorecase</b>	<b>ic</b>	Ignores distinction between upper- and lower-case while searching for regular expressions. The default is <b>noic</b> .
<b>lisp</b>	<b>lisp</b>	Removes the special meaning of <b>()</b> , <b>{}</b> , <b>[[</b> and <b>]]</b> and enables the <b>=</b> (formatted print) operator for <b>S</b> -expressions, so you can edit LISP programs. The default is <b>noisp</b> .
<b>list</b>	<b>list</b>	Displays text with tabs and the end of lines marked. Tabs display as <b>^I</b> and the end of lines as <b>\$</b> . The default is <b>noist</b> .
<b>magic</b>	<b>magic</b>	Treats the characters <b>.</b> (period), <b>[</b> , and <b>*</b> as special characters when scanning. In off mode, only the <b>( )</b> and <b>\$</b> retain special meanings. However, you can evoke special meaning in other characters by preceding them with a <b>\</b> (back slash). The default is <b>magic</b> .
<b>modeline</b>	<b>modeline</b>	Runs an editor command line if found in the first five or the last five lines of the file. An editor command line can be anywhere in a line. For the editor to recognize a command line, the line must contain a space or a tab followed by the string <b>ex:</b> or <b>vi:</b> . The command line is ended by a second <b>:</b> (colon). The editor tries to interpret any data between the first and second colon as editor commands. The default is <b>nomodeline</b> .
<b>number</b>	<b>nu</b>	Displays lines prefixed with their line numbers. The default is <b>nonu</b> .
<b>optimize</b>	<b>opt</b>	Speeds up the operation of terminals that lack cursor-addressing. The default is <b>noopt</b> .
<b>paragraphs</b>	<b>para =</b>	Defines <b>vi</b> macro names that start paragraphs. The default is <b>para = IPLPPPQPP LIpplpipnppb</b> . Single-letter <b>nroff</b> macros, such as <b>.P</b> , must include the space as a quoted character if respecting a paragraph.

---

Option	Abbreviation	Description
<b>partialchar</b>	<b>pc =</b>	Appears in the last display column where a double-wide character would not display completely. The default character is <b>-</b> (dash).
<b>redraw</b>	<b>redraw</b>	Simulates a smart work station on a dumb work station. The default is <b>none</b> .
<b>report</b>	<b>report =</b>	Sets the number of times you can repeat a command before a message is displayed. For subcommands that produce many messages, such as global subcommands, the messages are displayed when the command sequence completes. The default is <b>report = 5</b> .
<b>scroll</b>	<b>scr =</b>	Sets the number of lines to be scrolled when the user scrolls up or down. The default is <b>scroll = 12</b> .
<b>sections</b>	<b>sect =</b>	Defines to <b>vi</b> macro names that start sections. The default is <b>sect = NHHH HUuhsh + c</b> . Single-letter <b>nroff</b> macros, such as <b>.P</b> , must include the space as a quoted character if respecifying a paragraph.
<b>shell</b>	<b>sh =</b>	Defines the shell for <b>!</b> or <b>:! </b> commands. The default is the login shell.
<b>shiftwidth</b>	<b>sw =</b>	Sets the distance for the software tab stops used by <b>autoindent</b> , the shift commands ( <b>&gt;</b> and <b>&lt;</b> ), and the text input commands ( <b>Ctrl-D</b> and <b>Ctrl-T</b> ). The default is <b>sw = 8</b> .
<b>showmatch</b>	<b>sm</b>	Shows the matching open parenthesis ( <b>(</b> or open bracket <b>{</b> as you type the close parenthesis <b>)</b> or close bracket <b>}</b> . The default is <b>nosm</b> .
<b>slowopen</b>	<b>slow</b>	Postpones updating the display during inserts. The default is <b>noslow</b> .
<b>tabstops</b>	<b>ts =</b>	Sets distance between tab stops in a displayed file. The default is <b>ts = 8</b> .
<b>term</b>	<b>term =</b>	Sets the type of work station you are using. The default is <b>term = \$TERM</b> where <b>\$TERM</b> is the value of the shell variable <b>TERM</b> .

Option	Abbreviation	Description
<b>terse</b>	<b>terse</b>	Allows <b>vi</b> to display the short form of messages. The default is <b>noterse</b> .
<b>timeout</b>	<b>to</b>	Sets a time limit of two seconds on entry of characters. This limit allows the characters in a macro to be entered and processed as separate characters when <b>timeout</b> is set. To resume use of the macro, set <b>notimeout</b> . The default is <b>to</b> .
<b>warn</b>	<b>warn</b>	Displays a warning message before the <b>!</b> subcommand executes a shell command if it is the first time you issued a shell command after changes were made in the editing buffer but not written to a file. The default is <b>warn</b> .
<b>window</b>	<b>wi =</b>	Sets the number of lines displayed in one window of text. The default is dependent on the baud rate at which you are operating: 600 baud or less / 8 lines, 1200 baud / 16 lines, higher speeds / full screen minus one.
<b>wrapmargin</b>	<b>wm =</b>	Sets the margin for automatic word wrapping from one line to the next. The default is <b>wm = 0</b> . A value of zero turns off word wrapping.
<b>wrapscan</b>	<b>ws</b>	Allows string searches to wrap from the end of the editing buffer to the beginning. The default is <b>ws</b> .
<b>wraptype</b>	<b>wt =</b>	Determines how a line splits for word wrapping. The default is <b>general</b> . The default causes <b>vi</b> to insert a new line after a white space. Any characters following the insert point begin the new line. White space immediately before the inserted line is deleted. When Japanese Language Support is installed and <b>wraptype = general</b> , <b>vi</b> inserts a new line to wrap one character. One character forming closing punctuation is allowed past the right margin.
<b>writeany</b>	<b>wa</b>	Turns off the checks usually made before a <b>write</b> command. The default is <b>nowa</b> .

## Defining Macros

If you use a subcommand or sequence of subcommands frequently, you can create a macro that will issue the subcommand or sequence. To create a macro, enter the sequence of subcommands into an editing buffer named with a letter of the alphabet. When used, **a - z** overlay the contents of the buffer; **A - Z** append text to the previous contents of the buffer, allowing the building of a macro piece by piece.

To invoke the macro, enter `@x` where *x* is the letter name of the buffer. Enter `@@` to repeat the last macro you invoked.

## Mapping Keys

You can use the **map** command to map a keystroke to a subcommand or a sequence of subcommands. To map a key, enter `:map key subcommand` where *key* is the key you want to assign to a subcommand or sequence of subcommands, and *subcommand* is the subcommand or sequence of subcommands. For example, to map `@` to delete lines, enter:

```
:map @ dd
```

In this example, `@` is the key to which the subcommand is assigned and `dd` is the subcommand.

In the next example, a subcommand sequence is mapped to a key:

```
:map * {>}
```

The `*` is the key to which the subcommand sequence is assigned and `{>}` is the subcommand sequence. The `{` moves the cursor to the beginning of the paragraph and the `>` indents the paragraph to the next shiftwidth.

While in text input state, enter the command **:map!** to display a list of the current key mappings. To remove a key map, enter **:unmap string** or **:unmap! string**, where *string* is the string used after the **:map** command to set the key and subcommand sequence. For example, to remove key map for the previous example, enter:

```
:unmap * {>}
```

If function keys are defined for your terminal, you can include them in the **map** or **unmap** command by typing **Ctrl-V** before pressing the desired key. It is useful to define keys seldom used in editing with another key or function key (**F0 - F12**). For example, the **Shift**, **Ctrl**, or **Alt** can be defined in this way.

The abbreviation command **:ab** is similar to the **:map** command. For example, if you set the letter `s` equal to `Sam` with a **:map** command and then entered the following sentence,

```
s ate apples.
```

it would display as

```
Sam ate applesam.
```

The **:map** command does not recognize the difference between the macro `s` and the text `s`. With some restrictions, the **:ab** command does distinguish between text and a macro. Setting the macro in the previous example with the **:ab** command,

```
:ab s Sam
```

and typing the same sentence would result in the correct sentence, Sam ate apples. The **:ab** command only recognizes a macro when it is preceded by a blank space or tab character. If the following were entered,

```
Sam swims
```

the **ab** macro would translate the sentence as:

```
Sam Samwims
```

The abbreviated item can occupy more than one line. However, you must use the **ex** editor to remove a macro that contains a **Ctrl-M** (enter sequence). After entering **ex**, issue a **:unab abbreviation**, and return to **vi**. Remove macros without the **Ctrl-M** by using **:ab abbreviation**. After removing or changing abbreviations created with **:ab**, enter **:ab** to list all currently defined abbreviations.

If you use an IBM 3161 ASCII Display Station, IBM 3163 ASCII Display or IBM 3101 ASCII Display Station, the default key mapping of **vi** can cause you to lose data. To see the default mapping, issue a **:map** command. Specific problems arise with the **Esc-J** or **Shift-J** sequence. This sequence deletes any information from the current position of the cursor to the end of the file. To avoid problems, change this sequence using a **.exrc** file.

## Keeping a Customized Change

When you customize **vi** from the **vi** command line, the customization is in effect until you leave the editor. If you want to keep your assignments you must put the commands in the file **.exrc**. Each time you start the editor it reads this file. When you create the customization file, do not type the **:** (colon) before each command. The **:** is only required when entering commands on the command line. The following is an example of a **.exrc** file:

```
set ai aw
set wm=5
map @ dd
```

## Flags

- l Enters **vi** in LISP mode. In this mode, **vi** indents appropriately for LISP code and the **(, ), {, }, [[, and ]]** subcommands are modified to act appropriately for LISP.
- r [*file*] Recovers a file after an editor or system crash. If you do not specify *file*, **vi** displays a list of all saved files.

- 
- R**            Sets the **readonly** option to protect the *file* against overwriting.
  - t tag**        Edits the file containing the *tag* and positions the editor at its definition.
  - wnum**        Sets the default window size to *num*. This is useful when you use the editor over a low-speed line.
  - + [subcmd]**    Carries out the **ex** subcommand before editing begins. If you do not specify *subcmd*, the cursor is placed on the last line of the file.
  - y**            Overrides the maximum line setting of 1,048,560 with any value above 1024.

## Subcommands

In the following lists, **Esc** stands for pressing the **ESCAPE** key instead of pressing the **Enter** key.

### General Subcommand Syntax

*[named\_buffer]* *[operator]* *[number]* *object*

Surrounding square brackets indicate optional items.

- [named\_buffer]*    A temporary text storage area in memory.
- [operator]*        Specifies the subcommand or action; instructs **vi**.
- [number]*         A whole decimal value that specifies either the extent of the action, or a line address.
- object*            Specifies what to act on. This can be a text object (a character, word, sentence, paragraph, section, character string) or a text position (a line, position in the current line, screen position).

### Counts before Subcommands

You can prefix many subcommands with a number. The **vi** editor interprets this number in one of the following ways:

1. Go to line *number*:

```
5G
10z
```

2. Go to column *number*:

```
25i
```

3. Scroll *number* lines:

```
10Ctrl-D
10Ctrl-U
```

## Subcommands for Moving within the File

There are many commands that you can use to move within a file. They can be entered while **vi** is in the command state.

### *Movements within a Line*

- ← or **h** or **Ctrl-H**  
Moves the cursor one character to the left.
- ↓ or **j** or **Ctrl-J** or **Ctrl-N**  
Moves the cursor down one line (but it remains in the same column).
- ↑ or **k** or **Ctrl-P**  
Moves the cursor up one line (but it remains in the same column).
- or **l**  
Moves the cursor one character to the right.

### *Character Positioning within a Line*

- ^** Moves the cursor to the first nonblank character.
- 0** Moves the cursor to the beginning of the line.
- \$** Moves the cursor to the end of the line.
- fx** Moves the cursor to the next *x* character.
- Fx** Moves the cursor to the last *x* character.
- tx** Moves the cursor to one column before the next *x* character.
- Tx** Moves the cursor to one column after the last *x* character.
- ;** Repeat the last **f**, **F**, **t**, or **T** subcommand.
- ,** Repeat the last **f**, **F**, **t**, or **T** subcommand in the opposition direction.
- num|** Moves the cursor to the specified column.

### *Words, Sentences, Paragraphs*

- w** Moves the cursor to the next small word.
- b** Moves the cursor to the previous small word.
- e** Moves the cursor to the end of the small word
- W** Moves the cursor to the next big word.

- 
- B** Moves the cursor to the previous big word.  
**E** Moves the cursor to the end of a big word.

### ***Line Positioning***

- H** Moves the cursor to the top line on the screen.  
**L** Moves the cursor to the last line on the screen.  
**M** Moves the cursor to the middle line on the screen.  
**+** Moves the cursor to the next line at its first nonblank character.  
**-** Moves the cursor to the previous line at its first nonblank character.  
**Enter** Moves the cursor to the next line at its first nonblank character.

### ***Scrolling***

- Ctrl-U** Scrolls up one half screen.  
**Ctrl-D** Scrolls down one half screen.  
**Ctrl-F** Scrolls forward one screen.  
**Ctrl-B** Scrolls backward one screen.

### ***Searching for Patterns***

- [num]G** Places the cursor at line number *num* or to the last line if *num* is not specified.  
**/pattern** Places the cursor at the next line containing *pattern*.  
**?pattern** Places the cursor at the next previous line containing *pattern*.  
**n** Repeats last search for *pattern* in the same direction.  
**N** Repeats last search for *pattern* in the opposite direction.  
**/pattern/+ num** Places the cursor at the *num*th line after the line matching *pattern*.  
**?pattern?-num** Places the cursor at the *num*th line before the line matching *pattern*.  
**%** Finds the parentheses or brace that matches the one at the current cursor position.

### ***Moving to Sentences, Paragraphs, or Sections***

- ]]** Places the cursor at next section (or function if you are in LISP mode).
- [[** Places the cursor at previous section (or function if you are in LISP mode).
- (** Places the cursor at the beginning of the previous sentence (or the previous s-expression if you are in LISP mode).
- )** Places the cursor at the beginning of the next sentence (or the next s-expression if you are in LISP mode).
- {** Places the cursor at the beginning of the next paragraph (or at the next list if you are in LISP mode).
- }** Places the cursor at the beginning of the next paragraph, at the next section if you are in C mode, or at the next list if you are in LISP mode.

### ***Marking and Returning***

- "** Moves the cursor to the previous location of the current line.
- "** Moves cursor to the beginning of the line containing the pervious location off the current line.
- mx** Marks the current position with letter *x*.
- `x** Moves cursor to mark *x*.
- 'x** Moves cursor to the beginning of the line containing mark *x*.

### ***Adjusting the Screen***

- Ctrl-L** Clears and redraws the screen.
- Ctrl-R** Redraws the screen and eliminates blank lines marked with a @.
- z** Redraws the screen with the current line at the top of the screen.
- z-** Redraws the screen with the current line at the bottom of the screen.
- z.** Redraws the screen with the current line at the center of the screen.
- /pattern/z-** Redraws the screen with the line containing *pattern* at the bottom.
- znum** Makes the window *num* lines long.
- z+** Page Up.
- z^** Page Down.

---

**Ctrl-E**    Scrolls the window down one line.

**Ctrl-Y**    Scrolls the window up one line.

## Subcommands for Editing

Use the following subcommands to edit your text. Subcommands not followed by an \* (asterisk) should be entered in the text input state. These subcommands affect the text relative to the current cursor position. You return to the command state by pressing the **Esc** key.

### *Editing the File*

**a***text*    Inserts *text* after the cursor.

**A***text*    Adds *text* to the end of the line.

**C**        Changes rest of line (**c\$**).

**cc**        Changes a line.

**cw**        Changes a word.

**cw***text*    Changes word to *text*.

**D** \*        Deletes the rest of the line (**d\$**).

**dd** \*        Deletes a line.

**dw** \*        Deletes a word.

**i***text*    Inserts *text* before the cursor.

**I***text*    Inserts *text* before the first nonblank character in the line.

**J** \*        Joins lines.

**o**        Adds an empty line below the current line.

**O**        Adds an empty line above the current line.

**rx** \*        Replaces the current character with *x*. (Commands followed by \* do not enter the text input state.)

**R***text*    Overwrites characters with *text*.

**s** \*        Substitutes characters (**cl**).

**S** \*        Substitutes lines (**cc**).

**u** \*        Undoes the previous change.

**x** \*        Deletes a character.

- X \*       Deletes characters before cursor (**dh**).
- yw \*       Yanks a word into the **undo** buffer.
- yy \*       Yanks a line into the **undo** buffer.
- << \*       Shifts one line to the left.
- <L \*       Shifts all lines from the cursor to the end of the screen to the left.
- >> \*       Shifts one line to the right.
- >L \*       Shifts all lines from the cursor to the end of the screen to the right.
- ~ \*        Changes letters at cursor to opposite case.
- ! \*        Indents for LISP.

### *Corrections during Insert*

Use the following commands only while in text input state. They have different meanings in the command state.

- Ctrl-H**   Erases last character.
- Ctrl-W**   Erases last small word.  
**Note:** For more information on small words see “Character Sets with vi” in this section.
- \           Quotes the erase and kill characters.
- Esc**       Ends insertion, back to command state.
- Ctrl-?**   Interrupts, terminates insert or **Ctrl-D**.
- Ctrl-D**   Goes back to previous **autoindent** stop.
- ^**Ctrl-D**   Ends **autoindent** for this line only.
- 0**Ctrl-D**   Moves cursor back to left margin.
- Ctrl-V**   Enters nonprinting character.

### *Moving Text*

- p           Puts back text in the **undo** buffer after the cursor.
- P           Puts back text in the **undo** buffer before the cursor.
- "xp         Puts back text from the buffer *x*.

---

<b>"xd</b>	Deletes text into the buffer <i>x</i> .
<b>y</b>	Places the object that follows (for example, <b>w</b> for word) in the <b>undo</b> buffer.
<b>"xy</b>	Places the object that follows in the <i>x</i> buffer, where <i>x</i> is any letter.
<b>Y</b>	Places the line in the <b>undo</b> buffer.

### ***Restoring and Repeating Changes***

<b>u</b>	Undoes the last change.
<b>U</b>	Restores the current line if the cursor has not left the line since the last change.
<b>.</b>	Repeats the last change or increments the <b>"n p</b> command. <b>Note:</b> This subcommand is not meant for use with a macro. Enter <b>@@</b> to repeat a macro.
<b>"n p</b>	Retrieves the <i>n</i> th last delete of a complete line or block of lines.

### **Interrupting, Canceling, and Exiting vi**

<b>Q</b>	Enter <b>ex</b> editor in command state.
<b>:sh</b>	Runs a shell. You can return to <b>vi</b> by pressing <b>Ctrl-D</b> .
<b>!cmd</b>	Runs <i>cmd</i> , then returns.
<b>!!</b>	Repeats the last <b>!cmd</b> command.
<b>n!cmd</b>	Executes shell command <i>cmd</i> and replaces the number of lines specified by <i>n</i> with the output of <i>cmd</i> . If <i>n</i> is not specified, when the default is 1. If <i>cmd</i> expects standard input, the lines specified are used as input. Thus the command <b>!sort</b> can sort a paragraph.
<b>n!obj cmd</b>	Executes shell command <i>cmd</i> and replaces <i>n</i> with output of <i>cmd</i> . If <i>n</i> is not specified, the default is 1. If <i>cmd</i> expects standard input, the lines or <i>obj</i> specified is used as input.
<b>ZZ</b>	Exits <b>vi</b> , saving changes.
<b>:q</b>	Quits <b>vi</b> . If you have changed the contents of the editing buffer, <b>vi</b> displays a warning message and does not quit.
<b>:q!</b>	Quits <b>vi</b> , discarding the editing buffer.
<b>Esc</b>	Ends insert or ends an incomplete subcommand.
<b>Ctrl-L</b>	Redisplays a screen.
<b>Ctrl-R</b>	Redisplays the screen if <b>Ctrl-L</b> is the <b>→</b> key.

**Ctrl-?** Interrupts a subcommand.

## File Manipulation

**:e *file*** Edits *file*.

**:e!** Re-edits the current file and discards all changes.

**:e + *file***  
Edits *file* starting at the end.

**:e + *num***  
Edits *file* starting at line *num*.

**:e #** Edits the alternate file. The alternate file is usually the previous current file name. However, if changes are pending on the current file when a new file is called, the new file becomes the alternate file. This subcommand is the same as **Ctrl-A**.

**:n** Edits next file in the list entered on the command line.

**:n *files*** Specifies new list of files to edit.

**:r *file*** Reads the file into the editing buffer by adding new lines below the current cursor position.

**:r !*cmd*** Runs the shell command *cmd* and places its output in the file by adding new lines below the current cursor position.

**:ta *tag*** Edits a file containing *tag* at the location of *tag*.

**:w** Writes the editing buffer contents to the original file.

**:w *file*** Writes the editing buffer contents to the named *file*.

**:w! *file*** Overwrites *file* with the editing buffer contents.

**Ctrl-G** Shows current file name and line.

**Ctrl-A.** Edits the alternate file. The alternate file is usually the previous current file name. However, if changes are pending on the current file when a new file is called, the new file becomes the alternate file. This subcommand is the same as **:e #**.

## Related Information

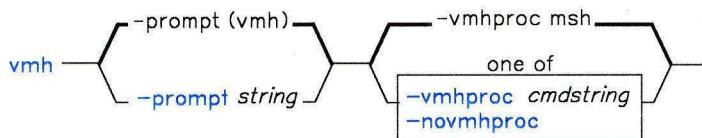
The following commands: “**ed**” on page 371 and “**ex**” on page 407.

# vmh

## Purpose

Invokes a visual interface for use with MH commands.

## Syntax



vmh — -help —

AJ2FL239

## Description

The **vmh** command is used to invoke a visual interface for use with MH commands. **vmh** is part of the MH (Message Handling) package and can be used with other MH and AIX commands.

The **vmh** command implements the server side of the MH window management protocol and maintains a split-screen interface to any program that implements the client side of the protocol.

**vmh** prompts for commands and sends them to the client side of the protocol. If the command produces a window of more than one screen's worth of output, **vmh** prompts the user for a subcommand.

## Subcommands

<b>SPACE</b>	Advances to the next screen.
[ <i>num</i> ] <b>ENTER</b>	Advances the specified number of lines. The default is one line.
[ <i>num</i> ] <b>y</b>	Goes back the specified number of lines. The default is one line.
[ <i>num</i> ] <b>d</b>	Advances ten times the specified number of lines. The default for <i>num</i> is 1, for a total of ten lines.

## vmh

---

[ <i>num</i> ] <b>u</b>	Goes back ten times the specified number of lines. The default for <i>num</i> is 1, for a total of ten lines.
[ <i>num</i> ] <b>g</b>	Goes to the specified line.
[ <i>num</i> ] <b>G</b>	Goes to the end of the window. If <i>num</i> is specified, this command acts like <b>g</b> .
<b>CTRL-L</b>	Refreshes the screen.
<b>h</b>	Displays a help message.
<b>q</b>	Ends output.

## Flags

<b>-help</b>	Displays help information for the command.
<b>-novmproc</b>	Runs the default <i>vmproc</i> directly, without the window management protocol.
<b>-prompt</b> <i>string</i>	Uses the specified string as the prompt.
<b>-vmhproc</b> <i>cmdstring</i>	Specifies the program which implements the client side of the window management protocol. The default is <b>msh</b> .

## Profile Entries

<b>Path:</b>	Specifies your <b>mhpath</b> .
<b>mshproc:</b>	Specifies the program used for the MH shell.

## Files

<code>\$HOME/.mh_profile</code>	The MH user profile.
---------------------------------	----------------------

## Related Information

The MH command “**msh**” on page 677.

The **mh-profile** file in *AIX Operating System Technical Reference*.

The “Overview of the Message Handling Package” in *Managing the AIX Operating System*.

# vrm2rtfont

---

## Purpose

Converts a standard AIX font file to RT rtx font format.

## Syntax

```
vrm2rtfont — aixfile — rtxfile —|
```

AJ2FL125

## Description

Before any of the standard fonts provided with the AIX Operating System can be used with the X-Windows program, they must be converted to the X-Windows' **rtx** font file-format. This command takes an AIX font file as input and converts the file to **rtx** format. This format can be used with X-Windows on RT displays.

## Examples

To convert the AIX font file **etc/vtm/itl1.9x20** to an **rtx** font file:

```
vrm2rtfont /etc/vtm/itl1.9x20 /usr/lpp/fonts/Itl14.500
```

Note how the output file name conforms to the **rtx** naming convention.

## Files

/etc/vtm	Contains AIX fonts.
/usr/lpp/fonts	Contains fonts for other programs.

## Related Information

The following command: "**vrm2rtfont**."

For more information on X-Windows, see *X-Windows*.

# vrconfig

---

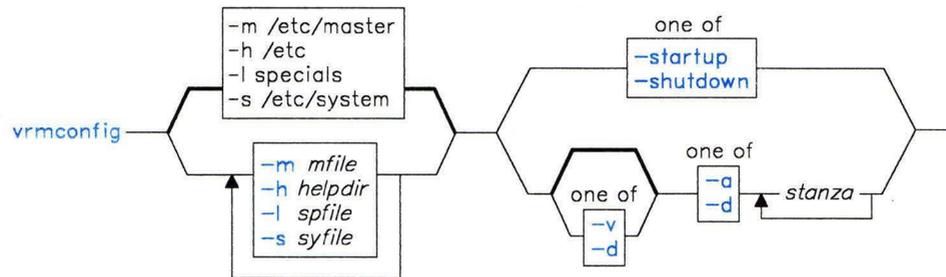
## vrconfig

---

### Purpose

Installs peripheral devices.

### Syntax



OL805400

### Description

The **vrconfig** command installs VRM device drivers. Normally, it runs automatically at each system startup. Its exit value is the number of errors it encountered. When auditing is on, an audit record of the type, **vrconfig** is created.

**Note:** Most users will never need to run this command from the command line.

The **vrconfig** command performs its operations through the **/dev/config** driver.

### Flags

- a stanza** Adds devices to a running system. **vrconfig** processes the specified *stanza* in **/etc/system** or the file specified with the **-s** flag.
- d stanza** Deletes a device from a running system. **vrconfig** processes the specified *stanza* in **/etc/system** or the file specified with **-s**. The special file list produced includes commands to remove relevant special files, since **vrconfig** assumes that the device has been removed from the configuration.
- h helpdir** Specifies the directory that contains the configuration helper programs. The default value is **/etc**.
- l spfile** Specifies the output special file list. The default value is **specials**.

- m** *mfile* Specifies the input master configuration file. The default value is **/etc/master**.
- s** *syfile* Specifies the input system configuration file. The default value is **/etc/system**.
- u** Causes only kernel-related configuration steps to be performed, that is, kernel driver installation calls. This flag may be used only with the **-a** or **-d** flags.
- v** Causes only VRM-related configuration steps to be performed, that is, **definedevice** calls. This flag may be used only with the **-a** or **-d** flags.
- shutdown** Causes all installed drives to be deleted for shutting down the system. Special files are not deleted, since the actual configuration is not considered changed.
- startup** Causes all defined devices to be configured in at system startup. (Any stanza that contains the attribute **noipl=true** is skipped.) For devices such as minidisks, which remain configured between system restarts, **vrmconfig** does not perform “once-only” configuration steps. It does not modify special files that already exist.

## Files

<i>/etc/configstatus</i>	Status file recording current configuration status.
<i>/etc/system</i>	Default system file.
<i>/etc/master</i>	Default master file.
<i>specials</i>	Default special file list.
<i>/etc</i>	Default directory containing configuration helper programs.
<i>/etc/vrmdd</i>	Directory containing VRM device driver modules.
<i>/vrm/vrmdd</i>	Directory containing VRM device driver modules.
<i>/etc/ddi</i>	Directory containing device specific information files.

## Related Information

The following command: “**config**” on page 194.

The **master** and **system** files in *AIX Operating System Technical Reference*.

*Installing and Customizing the AIX Operating System*.

# wall

---

# wall

---

## Purpose

Writes a message to all logged-in users.

## Syntax

`wall` `—`

OL805017

## Description

The **wall** command reads a message from standard input until it reaches an end-of-file character. It then sends the message to all logged-in users preceded by the following heading:

*Broadcast Message from user*

To override any protections other users have set up, you must be operating with superuser authority. Typically, **the superuser** uses **wall** to warn all users of an impending system shutdown.

**Note:** This command only sends messages to the local node.

## Files

`/dev/tty*`

## Related Information

The following commands: “**mesg**” on page 642, “**su**” on page 1026, and “**write**” on page 1225.

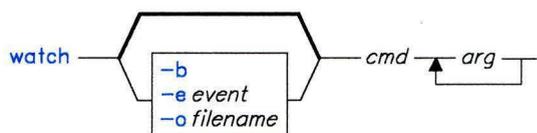
# watch

---

## Purpose

Observes and reports security-relevant actions on the system.

## Syntax



OL805487

## Description

The **watch** command allows a user with superuser authority to observe the actions of an untrustworthy program.

The **watch** command creates a channel to the audit device on which the events specified by **-e event** are recorded. If no *events* are specified, a default set of events is audited and appears on this device. The **watch** command reads these audit records and writes them to standard output.

The **watch** command executes the command specified by *cmd*, (with the arguments specified by *args*). If the *cmd* or any of its descendants perform an action specified by the *event*, the action is audited.

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

## Flags

- b** Specifies that audit records should be written out as binary records. Normally, the **watch** command first processes audit records through **auditpr**.
- e event** Specifies an event to be audited for the command. If no **-e** flag is specified, a list of events which contains all modifications of the trusted computing base is audited.

## **watch**

---

**-o filename** Specifies a file name for output. If the **-o** flag is not specified, audit records are written to standard output.

### **Files**

**/dev/audit** Used to monitor and enable the actions of descendent processes.

### **Related Information**

The discussion of trusted programs and the trusted computing base in *Managing the AIX Operating System*.

---

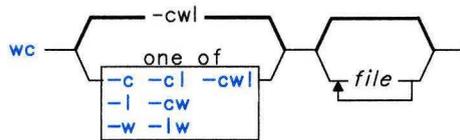
## WC

---

### Purpose

Counts the number of lines, words, and characters in a file.

### Syntax



OL805242

### Description

The **wc** command counts the number of lines, words, or characters in *file* or in the standard input if you do not specify any *files*. It writes the results to standard output. It also keeps a total count for all named files. A word is defined as a string of characters delimited by spaces, tabs, or new-line characters. **wc** counts lines, words, and characters by default.

When you specify more than one *file* on the command line, **wc** displays the name of the file along with the counts.

### Flags

- c Counts bytes.
- l Counts lines.
- w Counts words.

### Examples

1. To display the line, word, and character counts of a file:

```
wc chap1
```

This displays the number of lines, words, and characters in the file chap1.

2. To display only character and word counts:

```
wc -cw chap*
```

This displays the number of characters and words in each file whose name starts with chap, and displays the totals.

---

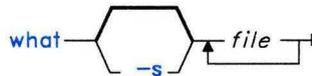
# what

---

## Purpose

Displays identifying information in files.

## Syntax



OL805295

## Description

The **what** command searches the named *files* for all occurrences of the pattern that **get** substitutes for the %Z% keyletter (see “Identification Keywords” on page 480). By convention, the value substituted is @(#).

**what** writes to standard output whatever follows the pattern up to but not including the first double quotation mark ("), greater than symbol (>), new-line character , backslash (\), or null character.

The **what** command is intended for use in conjunction with the **get** command, which automatically inserts the identifying information. You can also use **what** on files where the information is inserted manually.

## Flags

-s Searches for only the first occurrence of @(#).

## what

---

### Examples

Suppose that the file `test.c` contains a C program that includes the line:

```
char ident[ ] = "@(#)Test Program";
```

If you compile `test.c` to produce `test.o` and `a.out`, then the command:

```
what test.c test.o a.out
```

displays:

```
test.c: Test Program
test.o: Test Program
a.out:  Test Program
```

### Related Information

The following commands: “**get**” on page 477, and “**help**” on page 513.

The `scsfile` file in *AIX Operating System Technical Reference*.

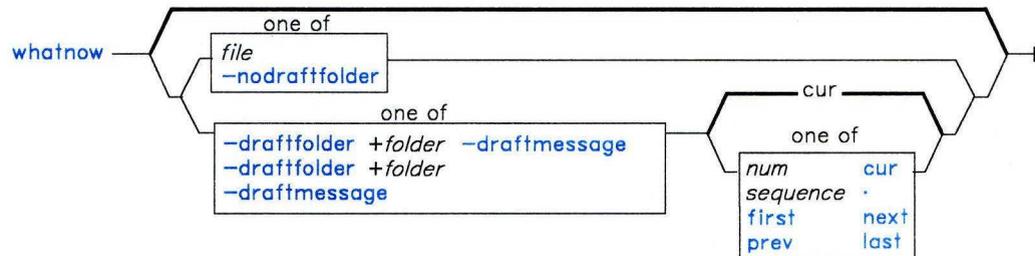
The discussion of SCCS in *AIX Operating System Programming Tools and Interfaces*.

# whatnow

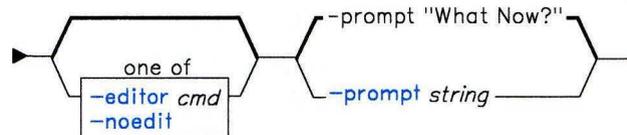
## Purpose

Invokes a prompting interface for draft disposition.

## Syntax



AJ2FL244



whatnow — -help —|

AJ2FL158

## Description

The **whatnow** command is the default program used by **comp**, **dist**, **forw**, and **repl** to prompt you for the disposition of messages. **whatnow** is part of the MH (Message Handling) package and can be used with other MH and AIX commands.

By default, **whatnow** invokes an editor and places the current draft message (or *file*) in the editing session. When you exit the editing session, **whatnow** prompts: What now? You can specify any of the **whatnow** subcommands, or you can press **Enter** to see a list of the subcommands. These subcommands enable you to re-edit the message, direct the disposition of the message, or end the processing of the **whatnow** command. The **whatnow** command continues to prompt you for subcommands until you specify **quit**.

## Subcommands

- display** [ *flags* ] Displays the message being acted upon (redistributed or replied to). For *flags*, you can specify any flag that is valid for the command serving as the *lproc*. (You can set a default **lproc:** entry in **\$HOME/.mh-profile**.) If you specify any flags that are not valid for *lproc*, **whatnow** does not pass the path name of the draft to the **lproc**.
- edit** [ *cmdstring* ] Re-edits the message using the specified editor. You can specify the editor and any valid flags to that editor. If you do not specify an editor, **whatnow** selects a default editor according to the information supplied in the MH profiles. You can define a default editor for re-editing in **\$HOME/.mh-profile**. If an editor is not defined for re-editing in your **.mh-profile**, **whatnow** invokes the editor used in the previous editing session.
- list** [ *flags* ] Displays the draft. For *flags*, you can specify any flag that is valid for the command serving as the *lproc*. (You can set a default **lproc:** entry in **\$HOME/.mh-profile**.) If you specify any flags that are not valid for *lproc*, **whatnow** does not pass the path name of the draft to *lproc*.
- push** [ *flags* ] Sends the message in the background. You can specify any valid flag for the **send** command.
- quit** [ **-delete** ] Ends the **whatnow** session. If you specify **-delete**, **whatnow** deletes the draft. Otherwise, **whatnow** stores the draft.
- refile** [ *flags* ] + *folder* Files the draft in the specified folder and supplies a new draft having the previously specified form. For *flags*, you can specify any flag that is valid for the command serving as the *fileproc*. (You can set a default **fileproc:** entry in **\$HOME/.mh-profile**.)
- send** [ *flags* ] Sends the message. You can specify any valid flags for the **send** command.
- whom** [ *flags* ] Displays the addresses to which the message would be sent. You can specify any valid flags for the **whom** command.

## Flags

- draftfolder** *+folder* Places the draft message in the specified folder. If you do not specify this flag, **whatnow** selects a default draft folder according to the information supplied in the MH profiles. You can define a default draft folder in **\$HOME/.mh-profile**. If **-draftfolder** *+folder* is followed by *msg*, *msg* represents the **-draftmessage** attribute.
- draftmessage** *msg* Specifies the draft message. You can use one of the following message references as *msg*:
- | <i>num</i>  | <i>sequence</i> | <b>first</b> |
|-------------|-----------------|--------------|
| <b>prev</b> | <b>cur</b>      |              |
| <b>next</b> | <b>last</b>     |              |
- The default draft message is **cur**.
- editor** *cmd* Specifies that *cmd* is the initial editor for composing or revising the message. If you do not specify this flag, **whatnow** selects a default editor or suppresses the initial edit, according to the information supplied in the MH profiles. You can define a default initial editor in **\$HOME/.mh-profile**.
- help** Displays help information for the command.
- nodraftfolder** Places the draft in the file *user-mh-directory/draft*.
- noedit** Suppresses the initial edit.
- prompt** *string* Uses *string* as the prompt. The default string is What now?

## Profile Entries

- Draft-Folder:** Sets your default folder for drafts.
- Editor:** Sets your default initial editor.
- fileproc:** Specifies the program used to refile messages.
- lasteditor-next:** Specifies the editor used after exiting *lasteditor*.
- lproc:** Specifies the program used to list the contents of a message.
- Path:** Specifies your *user-mh-directory*.
- sendproc:** Specifies the program used to send messages.
- whomproc:** Specifies the program used to determine the users to whom a message would be sent.

### Files

`$HOME/.mh-profile`    The MH user profile.  
`user-mh-directory/draft`    The draft file.

### Related Information

Other MH commands: “**comp**” on page 185, “**dist**” on page 336, “**forw**” on page 438, “**prompter**” on page 778, “**refile**” on page 817, “**repl**” on page 821, “**rmm**” on page 841, “**scan**” on page 871, “**send**” on page 893, “**whom**” on page 1222.

The **mh-alias**, **mh-format**, **mh-mail**, and **mh-profile** files in *AIX Operating System Technical Reference*.

The “Overview of the Message Handling Package” in *Managing the AIX Operating System*.

---

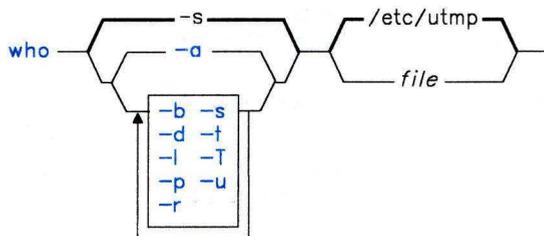
# who

---

## Purpose

Identifies the users currently logged in.

## Syntax



```
who am i —
```

AJ2FL120

## Description

The **who** command with no flags writes to standard output the login name, work station name, and date and time of login for all users currently on the system. Entering `who am i` displays your login name and work station name, and the date and time you logged on.

**Note:** The **who am i** command does not display the time of login when executed from a virtual terminal.

With flags, **who** can also display the elapsed time since line activity occurred, the process ID of the command interpreter (shell), logins, logoffs, restarts, and changes to the system clock, as well as other processes generated by the **init** process.

The general format of the output of **who** is as follows:

```
name [state] line time activity pid [location] [exit]
```

where:

- name is the user's login name.
- state indicates whether or not the line is readable by everyone (see the **-T** flag on page 1221).
- line is the name of the line as found in the directory **/dev**.
- time is the time the user logged in.

## who

---

- **activity** is the hours and minutes since activity last occurred on that user's line. A dot (.) here indicates line activity within the last minute. If the line has been quiet more than 24 hours or has not been used since the last system startup, the entry is marked as old.
- **pid** is the process ID of the user's shell.
- **location** is the location associated with this line as found in file **/etc/ports**. This file can contain information about where the work station is located, the telephone number of the dataset, the type of a direct-connected work station, and other related information.
- **exit** is the exit status of ended processes (see the **-d** flag on page 1220).

To obtain information, **who** normally examines **/etc/utmp**. If you specify another *file*, **who** examines the named *file* instead. This *file* will usually be **/usr/adm/wtmp**, which contains the history of all logins since the file was last created or **/etc/ilog**, which contains the history of invalid logins. Only someone operating with superuser authority or a member of the system group can examine **/etc/ilog**

**Note:** This command only identifies users on the local node.

## Flags

- a Processes **/etc/utmp** or the named *file* with all flags on.
- b Indicates the time and date of the most recent system startup. The **NLTIME** and **NLLDATE** environment variables control the format of the login time and date.
- d Displays all processes that have expired without being regenerated by **init**. The *exit* field appears for dead processes and contains the termination and exit values (as returned by **wait**) of the dead process. (This flag is useful for determining why a process ended.)
- l Lists only work stations not in use. The *name* field is **LOGIN** in such cases. Other fields are the same as for user entries except that the *state* field doesn't appear.
- p Lists any active process that is currently active and has been previously generated by **init**.
- r Indicates the current **run-level** of the process.
- s Lists only the *name*, *line*, and *time* fields. (This is the default; thus, **who** and **who -s** are equivalent.) The **NLTIME** environment variable controls the format of the time.
- t Indicates the last change to the system clock by the superuser using the **date** command. The **NLTIME** environment variable controls the format of the time.

- T Displays the *state* of the work station line and indicates who can write to that work station as follows:
  - + writable by anyone
  - writable only by the superuser or its owner
  - ? bad line encountered.
- u Displays the user name, work station name, login time, line activity, and process ID of each current user. The **NLTIME** environment variable controls the format of the login time.

## Examples

1. To display information about who is using the system:

```
who
```

This lists the user name, work station name, and login time of all users currently using the system.

2. To display your user name:

```
who am i
```

This displays the user name you typed when you logged in, the name of the work station you are using, and the time you logged in. Your login user name may be different from your current user name if you have used the **su** command.

3. To display a history of logins, logoffs, system startups, and system shutdowns:

```
who /usr/adm/wtmp
```

## Files

```
/etc/utmp  
/usr/adm/wtmp  
/etc/ports
```

## Related Information

The following commands: “**date**” on page 281, “**init**” on page 521, “**login**” on page 584, “**mesg**” on page 642, and “**su**” on page 1026.

The **wait** system call and the **ports** and **utmp** files in *AIX Operating System Technical Reference*.

“Overview of International Character Support” in *Managing the AIX Operating System*.

# whom

---

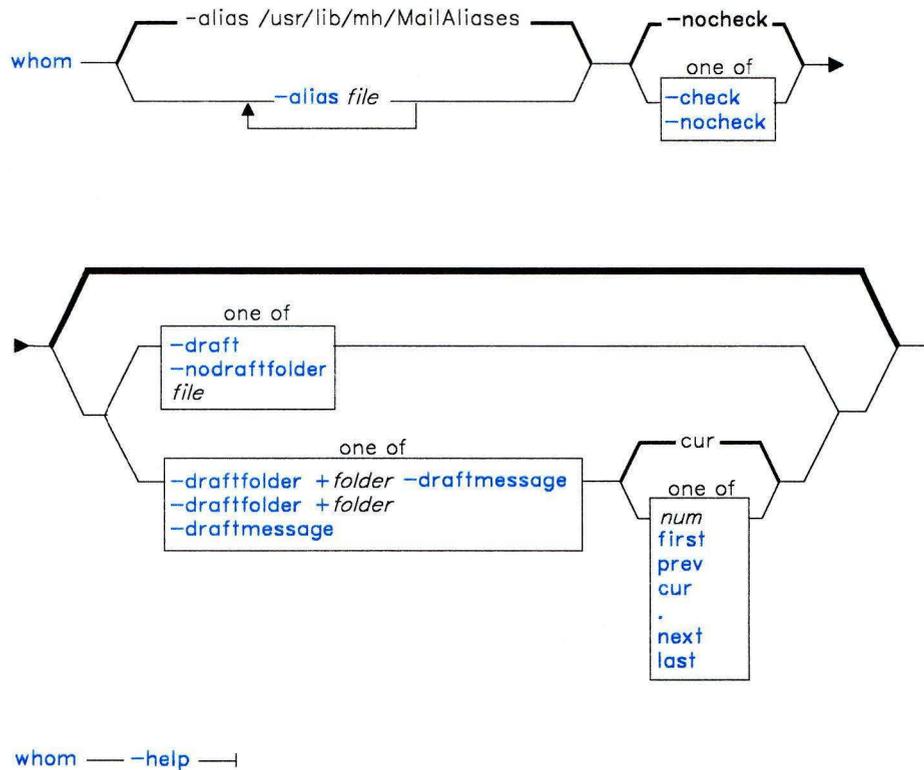
## whom

---

### Purpose

Lists the addresses of the proposed recipients of a message and verifies the addresses.

### Syntax



AJ2FL172

### Description

The **whom** command is used to expand the headers of a message into a set of addresses. **whom** is also used to verify that those addresses are valid. **whom** is part of the MH (Message Handling) package and can be used with other MH and AIX commands.

The message can reside in a draft folder or in a file. You can use one of the **-draft**, **-draftfolder**, **-draftmessage**, or **nodraftfolder** flags or the *file* argument to specify where the message resides.

If you want to verify the addresses, you must specify the **-check** flag.

## Flags

- alias *file*** Specifies that *file* is a mail alias file to be searched for aliases. The default alias file is **/usr/lib/mh/MailAliases**.
- check** Checks to see if the addresses are valid.
- draft** Uses the header information in the file *user-mh-directory/draft* if it exists.
- draftfolder + *folder*** Uses the header information of the draft message in the specified folder. If you do not specify this flag, **whom** selects a default draft folder according to the information supplied in the MH profiles. You can define a default draft folder in **\$HOME/.mh-profile**. If **-draftfolder + *folder*** is followed by *msg*, *msg* represents the **-draftmessage** attribute.
- draftmessage *msg*** Uses the header information in the specified draft message. You can use one of the following message references when specifying *msg*:
 

<i>num</i>	<i>sequence</i>	<b>first</b>
<b>prev</b>	<b>cur</b>	
<b>next</b>	<b>last</b>	

The default draft message is **cur**.
- help** Displays help information for the command.
- nocheck** Does not check to see if the addresses are valid. This is the default.
- nodraftfolder** Undoes the last occurrence of **-draftfolder + *folder***.

## Profile Entries

- Draft-Folder:** Sets your default folder for drafts.
- postproc:** Specifies the program used to post messages.

## whom

---

### Files

`$HOME/.mh_profile`      The MH user profile.

### Related Information

Other MH commands: “**ali**” on page 48, “**post**” on page 758, “**whatnow**” on page 1215.

The **mh-alias**, **mh-format**, **mh-mail**, and **mh-profile** files in *AIX Operating System Technical Reference*.

“Overview of the Message Handling Package” in *Managing the AIX Operating System*.

---

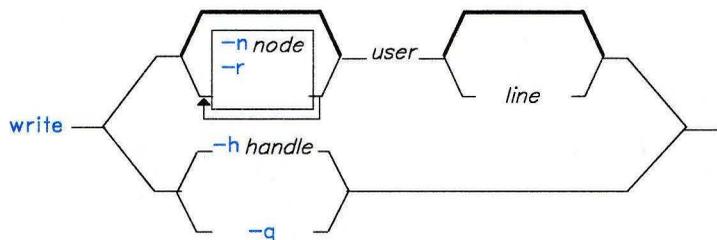
# write

---

## Purpose

Sends messages to other users on the system.

## Syntax



AJ2FL121

## Description

A common use of this command is to **converse** with another logged-in *user*. That is, each user alternately sends and receives short messages from the other work station. Long messages can be sent by first putting the complete message in a file and then redirecting that file as input to the **write** command.

For another *user* to receive your message, that user must be logged in and must not have refused message permission. When a person you are trying to reach is not logged in, you get the message `user not logged in`. When the person you are trying to reach has refused message permission, you get the message `write: permission denied`.

When you run the **write** command, it immediately sends the following message, along with an attention-getting sound (the ASCII BEL character) to the person whose login name you entered.

```

Message from yourid (tty $nn$ )
[date] . . .
  
```

After successful connection, **write** then sends two ASCII BEL characters to your work station to alert you that whatever you enter now is being sent to the other *user*. Sending continues until you press **Ctrl-D**, at which point **write** sends an end-of-text character to the other work station and exits.

## write

---

At this point, the other *user* can respond by sending a **write** message back. For this type of exchange, the following convention is useful: When you first **write** to others, wait for them to **write** back before sending any text. End a message with a signal such as 0 (over) to alert the other person to reply. Use 00 (over and out) when the conversation is finished.

When you **write** to a *user* logged in at more than one work station, **write** uses the first login instance found in file `/etc/utmp` as the message delivery point, and you get the message:

```
userid is logged on more than one place.  
You are connected to "work station".  
Other locations are:  
work station
```

You can contact this *user* at another location by specifying the *line*. *line* indicates to which work station (`tty00`, for example) the message should be sent.

Permission to **write** to another *user* is granted or denied by the other *user* with the **mesg** command. Some commands deny message permission while they are running to prevent interference with their output. A user with superuser authority can **write** to any work station regardless of the work station's message permission.

If Distributed Services is installed on your system, you can use the **write** command to converse with users on other nodes. You can identify a user on a remote node explicitly by using the **-n** flag or implicitly through entries in the file `/etc/wwwmachines`. This file contains a list of node IDs or nicknames of machines at which a user may be contacted. If you use the **-n** flag, **write** does not run through the list of machines named in `/etc/wwwmachines`.

The **write** command is also used by **qdaemon** to send messages to users on other nodes and to wait for replies. The contents of the message becomes the reply. Certain keywords in the reply message are recognized as having special meaning. If the user replies with the word `ok`, then the original **write** exits with a status of 0. If the user replies with the word `cancel`, then the **write** exits with a status of 1. If the reply contains the word `query`, then the message associated with the handle given is displayed.

## Flags

- n node** Specifies a remote node. The *node* field may be a nickname or a node-ID.
- h handle** Replies to a message sent by a utility or shell script using **write** with the reply option. The value to be used for *handle* is generated internally and supplied to the user in the text of the original message.

- r** Generates a message handle, places it in the message header, sends the message, and waits for a reply. This flag is used by **qdaemon** for operator messages and can be put in shell scripts. It is not used for interactive conversations. An exit status of 0 indicates that the reply was OK, a status of 1 indicates that the reply was cancel, and an exit status of 2 indicates that the user could not be contacted.
- q** Queries all messages awaiting replies from users on a node and displays them with their handles.

## Examples

1. To write a message to a user who is logged in:

```
write scottie  
I need to see you! Meet me in the computer room at 12:30.  
Ctrl-D
```

If your user ID is kirk and you are using work station tty3, scottie's work station displays:

```
Message from kirk tty3...  
I need to see you! Meet me in the computer room at 12:30.  
EOF
```

2. To hold a conversation:

```
write scottie  
Meet me in the computer room at 12:30.  
(o)
```

This starts the conversation. The (o) at the end stands for "over." It tells scottie that you are waiting for a response. *Do not* press **Ctrl-D** if you wish to continue.

Now Scottie replies by typing:

```
write kirk  
I'm running tests at 12:30. Can we meet at 3?  
(o)
```

And you might respond:

```
write scottie  
OK--the computer room at 3.  
(oo)
```

The (oo) stands for "over and out," telling Scottie that you have nothing more to say. If Scottie is also finished (oo), then you both press **Ctrl-D** to end the conversation.

## write

---

3. To write someone a prepared message:

```
write jay <message.text
```

This writes the contents of the file `message.text` to fred's work station.

4. To write to the person using a certain work station:

```
write - console
The printer in building 998 has jammed.
Please send help.
Ctrl-D
```

This writes the message to the person logged in at the work station `/dev/console`.

5. To send a message to user spuds at node partya:

```
write -n partya spuds
Your new tape has just arrived,
come see me to pick it up.
Thanks!
Ctrl-D
```

6. Here is an example of a message sent by **qdaemon**:

```
Message from mary on node 10813661 [ Aug 17 10:03:34 ] ...
[ Sent by qdaemon, use "write -h 6398492" to reply ]
```

```
Please insert tape number 5 into rmt0.
```

```
EOF
```

To reply in the affirmative enter:

```
write -h 6398492
ok
Ctrl-D
```

To reply in the negative enter:

```
write -h 6398492
cancel
Ctrl-D
```

**Note:** With the **-h** flag, there is no need to supply the node ID or user ID. This information is tracked with the handle.

## **Files**

<code>/etc/utmp</code>	Contains user and accounting information for the <b>who</b> , <b>write</b> , and <b>login</b> commands.
<code>/etc/wwwmachines</code>	Contains node IDs and nicknames of machines at which the users may be contacted.

## **Related Information**

The following commands: “**mesg**” on page 642, “**nroff**, **troff**” on page 709, “**pr**” on page 761, “**sh**” on page 913, “**wall**” on page 1208, and “**who**” on page 1219.

## writesrv

---

## writesrv

---

### Purpose

Allows Distributed Services users to send messages to and receive messages from a remote system.

### Syntax

writesrv —|

AJ2FL254

### Description

The **writesrv** command is a daemon that allows users in a Distributed Services environment to send messages to users on a remote system and receive responses from users on a remote system with the **write** command.

The **writesrv** utility receives incoming requests from a **write** command and creates a server process to handle the request. This server process communicates with the client process (**write**) and provides whatever services are requested.

To perform these services, the **writesrv** command creates an IPC queue using the key 0x203fe. All requests for service are sent as messages to this queue. The server also creates an IPC queue profile named **IBMWRTnode** where *node* is the node ID of the local system. This profile may be examined with the **ipctable** command but should not be modified.

### Related Information

The commands: “**write**” on page 1225 and “**ipctable**” on page 544.

The **msgget**, **msgctl**, **msgsnd**, and **msgrcv** system calls in *AIX Operating System Technical Reference*.

## wump

---

### Purpose

Plays the game Hunt the Wumpus.

### Syntax

`/usr/games/wump` —|

OL805232

### Description

A wumpus is a creature living in a cave with many rooms interconnected by tunnels. You move among the rooms trying to shoot the wumpus with an arrow and trying to avoid being eaten by the wumpus or falling into bottomless pits. There are also Super Bats that may pick you up and drop you in some randomly selected room. For moving among the rooms and shooting arrows, **wump** asks appropriate questions and follows your instructions.

After either you kill the wumpus, the wumpus eats you, or you fall into a Bottomless Pit, **wump** asks if you want a new game. To quit the game at any time, press **INTERRUPT** (**Alt-Pause**) or **END OF FILE** (**Ctrl-D**).

# xargs

---

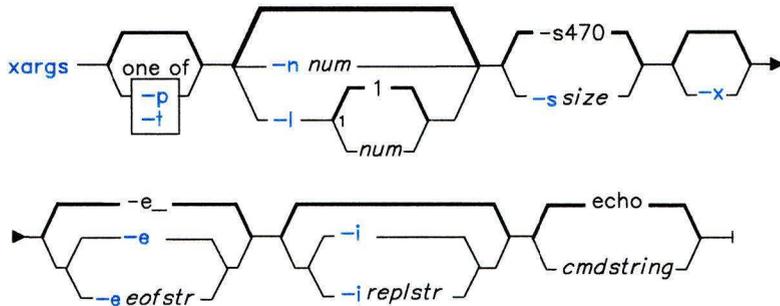
## xargs

---

### Purpose

Constructs argument lists and runs commands.

### Syntax



OL805298

### Description

The `xargs` command runs a command line. It constructs the command line by combining `cmdstring`, a string containing a command and its flags or parameters, with additional arguments read from standard input. It runs `cmdstring` as many times as necessary to process all input arguments. The default `cmdstring` is `echo`.

Arguments read from standard input are character strings delimited by one or more blanks, tabs, or new-line characters. You can embed a blank or a tab in arguments by preceding it with a `\` (backslash) or by putting it in quotation marks. `xargs` reads characters enclosed in single or double quotation marks as literals and removes the delimiting quotes. It always discards empty lines.

### Flags

`-e[eofstr]` Sets the logical end-of-file string to *eofstr*. `xargs` reads standard input until it encounters either an end-of-file character or the logical EOF string. If you do not specify the `-e` flag, the default *eofstr* is `_` (the underline character). If you specify `-e` with no *eofstr*, `xargs` interprets the underline character as a literal character rather than as an end-of-file marker.

- 
- i[replstr]** Takes an entire line as a single argument and inserts it in each instance of *replstr* found in *cmdstring*. A maximum of five arguments in *cmdstring* can each contain one or more instances of *replstr*. **xargs** discards blanks and tabs at the beginning of each line. The argument constructed cannot be larger than 255 bytes. The default *replstr* is `{}`. This flag also turns on the **-x** flag.
- l[num]** Runs *cmdstring* with the specified *num* of nonempty argument lines read from standard input. The last invocation of *cmdstring* can have fewer argument lines if fewer than *num* remain. A line ends with the first new-line character unless the last character of the line is a blank or a tab. A trailing blank or tab indicates a continuation through the next non-empty line. The default *num* is 1. This flag turns on the **-x** flag.
- nnum** Executes *cmdstring* using as many standard input arguments as possible, up to a maximum of *num*. The **xargs** command uses fewer arguments if their total size is greater than the number of bytes specified by the **-ssize** flag described below. It also uses fewer arguments for the last invocation if fewer than *num* arguments remain. When **-x** is present, each *num* argument must fit the *size* limitation specified by **-x**.
- p** Asks whether or not to run *cmdstring*. It displays the constructed command line, followed by a `? . . .` prompt. Press `y` to run the *cmdstring*. Any other response causes **xargs** to skip that particular invocation of *cmdstring*. You are asked about each invocation.
- ssize** Sets the maximum total size of each argument list. *size* must be a positive integer less than or equal to 470. The default *size* is 470 bytes. Note that the byte count for *size* includes one extra byte for each argument and the number of bytes in the command name.
- t** Echoes the *cmdstring* and each constructed argument list to file descriptor 2 (usually standard error).
- x** Stops running **xargs** if any argument list is greater than the number of bytes specified by the **-ssize**. This flag is turned on if you specify either the **-i** or **-l** flags. If you do not specify **-i**, **-l**, or **-n**, the total length of all arguments must be within the *size* limit.

**Note:** The **xargs** command ends if it cannot run *cmdstring* or if it receives a return code of -1. When *cmdstring* calls a shell procedure, the shell procedure should explicitly **exit** with an appropriate value to avoid accidentally returning -1. (See “**sh**” on page 913.)

## Examples

1. To use a command on files whose names are listed in a file:

```
xargs lint -a <files1 *
```

## xargs

---

If `cfiles` contains the text:

```
main.c readit.c
gettoken.c
putobj.c
```

then **xargs** constructs and runs the command:

```
lint -a main.c readit.c gettoken.c putobj.c
```

Each shell command line can be up to 470 bytes long. If `cfiles` contains more file names than fit on a single line, then **xargs** runs the **lint** command with the file names that fit. It then constructs and runs another **lint** command using the remaining file names. Depending on the names listed in `cfiles`, the commands might look like:

```
lint -a main.c readit.c gettoken.c . . .
lint -a getisx.c getprp.c getpid.c . . .
lint -a fltadd.c fltmult.c fltdiv.c . . .
```

This is not quite the same as running **lint** once with all the file names. The **lint** command checks cross-references between files. However, in this example it cannot check between `main.c` and `fltadd.c`, or between any two files listed on separate command lines.

For this reason you may want to run the command only if all the file names fit on one line. Tell **xargs** this by using the **-x** flag:

```
xargs -x lint -a <cfiles
```

If all the file names in `cfiles` do not fit on one command line, then **xargs** displays an error message.

2. To construct commands that contain a certain number of file names:

```
xargs -t -n2 diff <<end
starting chap1 concepts chap2 writing
chap3
end
```

This constructs and runs **diff** commands that contain two file names each (**-n2**):

```
diff starting chap1
diff concepts chap2
diff writing chap3
```

The **-t** flag tells **xargs** to display each command before running it so that you can see what is happening. The `<<end` and `end` define a “Here Document,” which uses the text entered before the `end` line as standard input for the **xargs** command. For more details, see “Inline Input Documents” on page 928.

3. To insert file names into the middle of commands:

```
ls | xargs -t -i mv {} {}.old
```

This renames all files in the current directory by adding `.old` to the end of each name. The `-i` tells **xargs** to insert each line of the `ls` directory listing where a `{}` appears. If the current directory contains the files `chap1`, `chap2`, and `chap3`, then this constructs the commands:

```
mv chap1 chap1.old
mv chap2 chap2.old
mv chap3 chap3.old
```

4. To run a command on files that you select individually:

```
ls | xargs -p -n1 ar r lib.a
```

This allows you to select files to add to the library `lib.a`. The `-p` flag tells **xargs** to display each `ar` command it constructs and ask if you want to run it. Type `y` and press **Enter** to run the command. Press **Enter** alone if you do not want to run it.

## Related Information

The following command: “**sh**” on page 913.

# xdbx

---

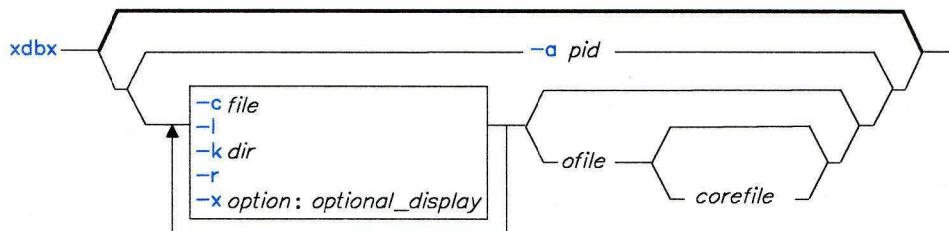
## xdbx

---

### Purpose

Provides an overlaying X-Window application for the dbx symbolic debugger.

### Syntax



AJ2FL128

### Description

The `xdbx` command is a user interface for the `dbx` source level debugger that provides a multi-window environment for interactive debugging of C, Pascal, and FORTRAN programs. All `dbx` flags, subcommands, and options can be used with `xdbx`.

**Note:** The `-x` flag is available with `xdbx` but not with `dbx`.

You must start X-Windows before using the `xdbx` command. The `xdbx` debugger displays up to three separate menu windows depending on the debugging mode. You select a debugging mode from `dbx` or `xdbx` using the `dbx` subcommand `set`.

### Flags

**-x option: (optional display)** Runs the display portion of `xdbx` on a remote host. Where `option` is the name of the remote host, and the `optional display` is a small integer designating the display. The default is 0.

### Related Information

The following command: “`dbx`” on page 284.

A discussion of how to debug programs in *AIX Operating System Programming Tools and Interfaces*.

---

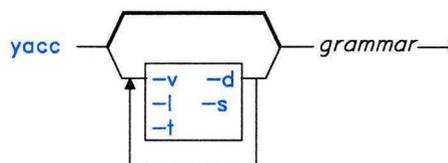
## yacc

---

### Purpose

Generates a LR(1) parsing program from input consisting of a context-free grammar specification.

### Syntax



OL805300

### Description

The **yacc** command converts a context-free grammar into a set of tables for a simple automaton that executes an LR(1) parsing algorithm. The grammar can be ambiguous; specified precedence rules are used to break ambiguities.

You must compile the output file, **y.tab.c**, with a C Language compiler to produce a function **yyparse**. This function must be loaded with the lexical analyzer function **yylex**, as well as **main** and **yyerror**, an error-handling routine (you must provide these routines). The **lex** command is useful for creating lexical analyzers usable by **yacc**.

For more detailed discussion of **yacc** and its operations, see *AIX Operating System Programming Tools and Interfaces*.

---

#### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

### Flags

- d** Produces the file **y.tab.h**. This contains the **#define** statements that associate the **yacc**-assigned token codes with your token names. This allows source files other than **y.tab.c** to access the token codes by including this header file.

## yacc

---

- l Does not include any **#line** constructs in **y.tab.c**. Use this only after the grammar and associated actions are fully debugged.
- s Breaks the **yyparse** function into several smaller functions. Since its size is somewhat proportional to that of the grammar, it is possible for **yyparse** to become too large to compile, optimize, or execute efficiently.
- t Compiles run-time debugging code. By default, this code is not included when **y.tab.c** is compiled. However, the run-time debugging code is under the control of **YYDEBUG**, a global variable for the **cc** command preprocessor. If **YYDEBUG** has a nonzero value, the C compiler (**cc**) includes the debugging code, whether or not the **-t** flag was used. Without compiling this code, **yyparse** will have a faster operating speed.
- v Prepares the file **y.output**. It contains a readable description of the parsing tables and a report on conflicts generated by grammar ambiguities.

## Files

y.output	
y.tab.c	
y.tab.h	Definitions for token names.
yacc.tmp,	
yacc.debug	Temporary file.
yacc.acts	Temporary file.
/usr/lib/yaccpar	Parser prototype for C programs.

## Related Information

The following command: “**lex**” on page 562.

The description of **yacc** in *AIX Operating System Programming Tools and Interfaces*.

---

# ypbind

---

## Purpose

Allows Yellow Pages client processes to communicate with the YP server.

## Syntax

`/etc/ypbind`

A5AC5011

## Description

The **ypbind** command remembers information that allows client processes on a single node to communicate with some **ypserv** process. The **ypbind** daemon must run on all machines using YP services, both YP servers and clients. The **ypbind** daemon is typically activated at system startup time from `/etc/rc.nfs`, if that file contains the appropriate entry.

The information handled by **ypbind** for a particular server is called a *binding*. A binding associates a domain name with both the Internet address of a YP server and the server's port at which the **ypserv** process is listening for service requests.

When a request for an unbound domain comes in, the **ypbind** process broadcasts on the network in order to find a **ypserv** process that serves maps within that domain. When a domain is bound by a particular **ypbind** process, that binding is given to every client process on the node. The **ypwhich** command can be used to query the **ypbind** process on the local node or a remote node for the binding of a particular domain.

Bindings are verified before they are given out to a client process. If **ypbind** is unable to communicate with the **ypserv** process to which it is bound, it marks the domain as unbound. Then it informs the client process that the domain is unbound and tries again to bind the domain.

Requests for an unbound domain fail immediately. In general, a bound domain is marked as unbound when the node running **ypserv** crashes or is overloaded. Then **ypbind** attempts to bind to any YP server available on the network.

The **ypbind** daemon accepts requests to set its binding for a particular domain. Such requests are typically generated by the YP subsystem itself.

## ypbind

---

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

## File

/etc/rc.nfs

## Related Information

The following commands: “**ypcat**” on page 1241, “**ypmatch**” on page 1245, “**yppush**” on page 1252, “**ypserv**” on page 1256, “**ypset**” on page 1254, and “**ypwhich**” on page 1258.

The section on Yellow Pages in *Managing the AIX Operating System*.

The Yellow Pages library section in *AIX Operating System Technical Reference*.

---

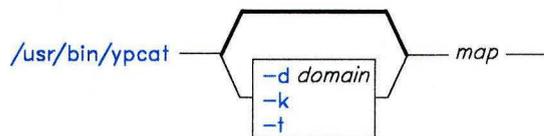
# ypcat

---

## Purpose

Displays values in a Yellow Pages data base.

## Syntax



```
/usr/bin/ypcat -x
```

A5AC5016

## Description

The **ypcat** command displays values in a Yellow Pages map specified by *map*, which can be either a map name or a map nickname. No YP server is specified since **ypcat** uses the YP network services.

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

## Flags

- d** Specifies a *domain* other than the default domain.
- k** Displays the keys for those maps in which the values are null or the key is not part of the value.
- t** Inhibits translation of *map* nickname to a map name. For example, the command sequence `ypcat -t passwd` fails because there is no map named *passwd*. However, the sequence `ypcat passwd` works because the map nickname *passwd* is translated to the map name *passwd.byname*.

## ypcat

---

- x Displays the map nickname table. The table lists valid nicknames and shows the map name associated with each nickname.

### Related Information

The following commands: “**ypmatch**” on page 1245 and “**domainname**” on page 340.

---

# ypinit

---

## Purpose

Builds and installs a Yellow Pages (YP) data base on the YP master server and YP slave servers.

## Syntax

```
/etc/yp/ypinit -m   
/etc/yp/ypinit -s mastername   

```

A5AC5003

## Description

The **ypinit** command sets up a Yellow Pages data base on a YP master server or YP slave server. Only users with superuser authority can use **ypinit**.

With the **-m** option, **ypinit** initially sets up a YP master server which functions as the master for all maps in the data base. After initialization you can change the association of maps to masters. The **ypinit -m** command invokes the **make** procedure, which follows the instructions in **/etc/yp/Makefile**. By default, it uses the ASCII system files as input files to the data bases being created. See the Files section later in this discussion for a list of the default files **ypinit** uses to create the data base. The files used to build the data bases should be in their full-length form rather than in the abbreviated form used on client machines.

With the **-s** option, the YP data base on a YP slave server is set up by copying a data base from the YP server specified by the *mastername* parameter (the machine's host name). The YP server from which the data base is being copied can be the YP master server, or a YP slave server whose data base is up to date and stable.

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

## Flags

**-m** Specifies the local host as the YP master server, and installs the YP data base of maps.

## ypinit

---

**-s *mastername*** Sets up a YP slave server by installing a copy of the master data base from the host specified by the *mastername* argument.

### Files

*/etc/passwd*  
*/etc/group*  
*/etc/hosts*  
*/etc/networks*  
*/etc/protocols*  
*/etc/netgroup* (if set up)  
*/etc/rpc*  
*/etc/services*  
*/etc/ethers* (if set up)  
*/etc/networks* (if set up)  
*/usr/lib/aliases* (if set up)

### Related Information

The following commands: “**makedbm**” on page 632 and “**ypxfr**” on page 1260.

The Yellow Pages section in *Managing the AIX Operating System*.

---

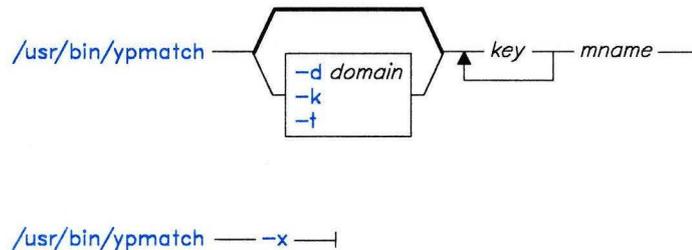
# ypmatch

---

## Purpose

Displays the value of one or more keys from a Yellow Pages map.

## Syntax



A5AC5018

## Description

The **ypmatch** command displays the values associated with one or more keys from the Yellow Pages (YP) map specified by *mname*. The *mname* parameter may be either a mapname or a map nickname.

When multiple keys are specified by the *key* parameter, the same map is searched for all the specified keys. Pattern matching is not done for *keys*, so the exact string, including capitalization, must be given.

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

## Flags

- d** Specifies a *domain* other than the default domain.
- k** Displays the key itself, followed by a colon, before displaying the value of the key. This flag is useful if the keys and their values are not identical, or if you specify so many keys that the output would be difficult to read.

## ypmatch

---

- t        Inhibits translation of *mname* to a map name. For example, the command sequence `ypmatch -t zippy passwd` fails because there is no map named *passwd*. However, the sequence `ypmatch zippy passwd` works because *passwd* is translated to the mapname *passwd.byname*.
- x        Displays the map nickname table. The table lists valid nicknames and shows the mapname associated with each nickname.

### Related Information

The following commands: “**domainname**” on page 340 and “**ypcat**” on page 1241.

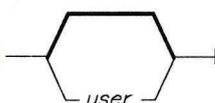
# yppasswd

---

## Purpose

Establishes or changes your Yellow Pages password.

## Syntax

`/usr/bin/yppasswd` 

A5AC5001

## Description

The **yppasswd** command establishes or changes your Yellow Pages password entry. Your YP password may be different from the login password you use on your own machine.

When you enter the **yppasswd** command you are prompted for your old YP password. If you do not have an old password, press **Enter**. You then receive the prompt for your new password. If you do have an old YP password, you must type it in correctly in order to be prompted for a new password.

After entering your new password once, you are prompted for it a second time. The first and second entries must match in order for the new password to be validated.

Passwords should be at least four characters long, or six characters long if you use only upper case letters or only lower case letters. Superuser authority is required to change the YP password of another user.

**Note:** The YP password update protocol passes all the initial information to the server in a single RPC call. If you enter your old password incorrectly, you are not informed until after you enter your new password.

The **yppasswd** daemon must be running on your YP server in order for your new password to take effect.

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

# yppasswd

---

## Files

/etc/yp/Makefile  
/etc/passwd

## Related Information

The following commands: “**passwd**” on page 735 and “**yppasswdd**” on page 1249.

The section on Yellow Pages in *Managing the AIX Operating System*.

The **/etc/passwd** file format in *AIX Operating System Technical Reference*.



# yppasswdd

---

## Flag

- m** Runs the **make** command using **Makefile** in the **/etc/yp** directory in order to update the changed password in the YP password data base. Any arguments that follow the flag are passed to the **make** command.

## Files

`/etc/yp/Makefile`  
`/etc/passwd`

## Related Information

The following command: “**yppasswd**” on page 1247.

The section on Yellow Pages in *Managing the AIX Operating System*.

The `/etc/passwd` file format in *AIX Operating System Technical Reference*.

---

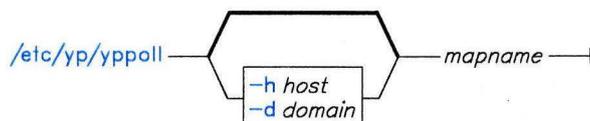
# yppoll

---

## Purpose

Displays the version of a Yellow Pages map located at a Yellow Pages server.

## Syntax



A5AC5013

## Description

The **yppoll** command queries the **ypserv** process for the order number and name of the host that is the YP master server for the map named by *mapname*.

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

## Flags

- d** Specifies a *domain* other than the default domain.
- h** Queries the **ypserv** process at the specified *host* for the map parameters. If **host** is not specified, the YP server for the local host is used.

## Related Information

The following command: “**ypwhich**” on page 1258.

# yppush

---

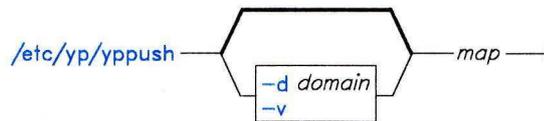
## yppush

---

### Purpose

Forces the distribution of a changed Yellow Pages map.

### Syntax



A5AC5012

### Description

The **yppush** command copies a new version of a Yellow Pages map from the YP master server to YP slave servers. It is normally run only on the YP master server from **/etc/yp/Makefile** after the data bases on the YP master server are changed.

The **yppush** command constructs a list of the YP server hosts by reading the YP map **ypservers** within the specified *domain*. The **ypservers** map contains the names of the machines which maintain YP maps. A request to transfer a map is sent to each YP server, along with the information needed to call back the **yppush** process.

---

#### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

### Flags

- d** Specifies a *domain* other than the default domain.
- v** Verbose. Causes messages to be displayed when each server is called, and displays one message for each response. Only error messages are displayed if this flag is omitted.

## Files

*/etc/yp/domainname/ypservers.dir*  
*/etc/yp/domainname/ypservers.pag*

## Related Information

The following commands: “**makedbm**” on page 632 and “**ypxfr**” on page 1260.

## ypset

---

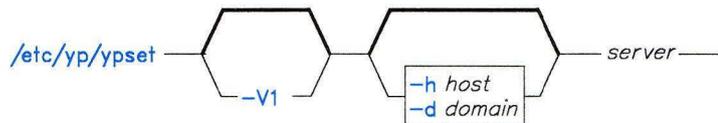
## ypset

---

### Purpose

Directs the **ypbind** daemon to a particular server.

### Syntax



A5AC5009

### Description

The **ypset** command directs the **ypbind** daemon to get YP services for the specified *domain* from the **ypserv** daemon running on the specified *server*. The binding set by **ypset** is tested by **ypbind** when a client process tries to get a binding for the *domain*. If a binding is invalid (*server* is down or is not running **ypserv**), **ypbind** keeps trying to bind for the same domain.

The **ypset** command can be used to bind a client node that is not on a broadcast network, or to bind a client node on a broadcast network that is not running a YP host. The command can also be used for debugging Yellow Pages client applications, such as when a YP map only exists on a single YP server host.

If several hosts on the local network are supplying YP services, **ypbind** can rebind to another host while you are attempting to find out if the **ypset** operation succeeded. Then you might see the response *host2* after typing `ypset host1` and `ypwhich`. Such a response is simply a function of the the YP subsystem's attempt to balance its load among the available YP servers. The response occurs when *host1* is not running **ypserv** or is overloaded, and *host2* is running **ypserv** and is not overloaded.

The *server* parameter is the YP server to which a machine or node binds. It can be a name or an IP address. If a name is specified, **ypset** tries to use YP services to resolve the name to an IP address. The name can be resolved to an address only if the node has a current valid binding for the *domain*.

---

**Japanese Language Support Information**

If Japanese Language Support is installed on your system, this command is not available.

---

**Flags**

- d        Specifies a *domain* other than the default domain.
- h        Sets **ypbind** on *host* instead of locally.
- V1      Binds *server* for Version 1 YP protocol.

**Related Information**

The following commands: “**domainname**” on page 340, “**ypwhich**” on page 1258, “**ypserv**” on page 1256, and “**ypbind**” on page 1239.

## ypserv

---

## ypserv

---

### Purpose

Looks up information in the local data base of Yellow Pages maps.

### Syntax

`/usr/etc/ypserv` —|

A5AC5010

### Description

The **ypserv** daemon is typically activated at system startup time from `/etc/rc.nfs`, if that file contains the appropriate entry. The **ypserv** daemon runs only on Yellow Pages server machines that have a complete YP data base.

The **ypserv** daemon's primary function is to look up information in its local data base of YP maps. The operations performed by **ypserv** are defined for programmers by the `rpcsvc/yp_prot.h` header file, and for network implementors by the YP *protocol specification*. Communication with **ypserv** is by means of Remote Procedure Calls.

There are four YP lookup operations that are performed on a specified map within some YP domain: **Match**, **Get\_first**, **Get\_next**, and **Get\_all**. The **Match** operation takes a key and returns the associated value. The **Get\_first** operation returns the first key-value pair from the map, and **Get\_next** returns a certain number of the remaining key-value pairs (as specified in the program). The **Get\_all** operation ships the entire YP map to a requestor in response to a single RPC request. These lookup operations are supplied as the following C-callable functions in `/lib/libc`: **yp-match**, **yp-first**, **yp-next**, and **yp-all**.

There are two other operations, **Get\_order\_number** and **Get\_master\_name**, that supply information about a map instead of map entries. Both order number and master name actually exist in the map as key-value pairs, but the server does not return either of them through the normal lookup functions. However, they will be visible if you examine the map with the **makedbm** command. **Get\_order\_number** and **Get\_master\_name** are supplied as the following C-callable functions in `/lib/libc`: **yp-order** and **yp-master**.

Log information is written to the file `/etc/yp/ypserv` if it exists when **ypserv** starts running.

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

## Files

/etc/rc.nfs  
/etc/yp/ypserv.log

## Related Information

The following commands: “**ypbind**” on page 1239, “**ypcat**” on page 1241, “**ypmatch**” on page 1245, “**yppush**” on page 1252, “**ypset**” on page 1254, “**ypwhich**” on page 1258, and “**ypxfr**” on page 1260.

The section on the Yellow Pages in *Managing the AIX Operating System*.

The Yellow Pages section in *AIX Operating System Technical Reference*.

# ypwhich

---

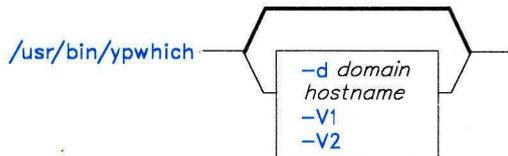
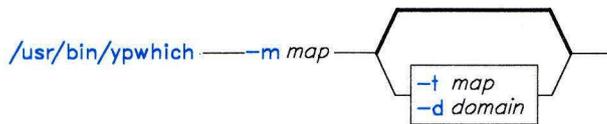
## ypwhich

---

### Purpose

Displays the name of the host machine acting as the Yellow Pages server or as a YP map server.

### Syntax



A5AC5008

### Description

The **ypwhich** command shows which YP server supplies Yellow Pages services to a YP client, or shows which server is the YP master server for a map. With no flags or arguments specified, the **ypwhich** command displays the name of the YP server for the local machine. When a particular machine is specified by **hostname**, that machine is queried for the name of the YP server it is using.

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

## Flags

- d** Specifies a *domain* other than the default domain.
- m** Finds the master YP server for a map. No *hostname* can be specified if **-m** is selected. The *map* parameter can be a mapname or a nickname for a map. A list of available maps is displayed when *map* is omitted.
- t** Inhibits nickname translation. This flag is used if *map* is identical to a nickname.
- V1** Displays name of server using Version 1 YP protocol.
- V2** Displays name of server using Version 2 YP protocol.
- x** Displays the map nickname table. The table lists valid nicknames and shows the mapname associated with each nickname.

## Related Information

The following commands: “**ypset**” on page 1254 and “**yppoll**” on page 1251.

# ypxfr

---

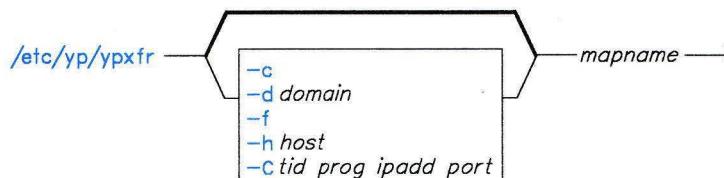
## ypxfr

---

### Purpose

Transfers a Yellow Pages map to the local host machine.

### Syntax



A5AC5006

### Description

The **ypxfr** command moves a Yellow Pages map to the local host by:

- Creating a temporary map in **/etc/yp/domain** (which must already exist)
- Filling the map by enumerating its entries
- Fetching and loading the map parameters (order number and server)
- Deleting any old versions of the map
- Assigning *mapname* to the temporary map, making it the new map.

If used interactively, **ypxfr** sends output to the user's terminal. If invoked without a controlling terminal, **ypxfr** appends its output to the file **/etc/yp/ypxfr.log** (if the file exists). The **ypxfr** command is most often invoked from **crontab** or by the **ypserv** daemon.

To maintain consistent information between servers, **ypxfr** should be used periodically for every map in the YP data base. Some maps change more frequently than others, and therefore should be updated more often. For example, the services name map can change once every few months, whereas the mail aliases and password name maps can change several times a day.

You can use a **crontab** entry to perform periodic updates automatically. You can also group commands together in a shell script to update several maps at once. For useful examples, refer to **ypxfr-1perd**, **ypxfr-2perd**, and **ypxfr-1perh** in the **/etc/yp** directory.

---

### Japanese Language Support Information

If Japanese Language Support is installed on your system, this command is not available.

---

## Flags

- c Prevents a "clear current map request" from being sent to the local **ypserv** process. Use this flag if **ypserv** is not running locally at the time you are running **ypxfr**.
- C This option is *only* for use by **ypserv**. The parameters *tid*, *prog*, *ipadd*, and *port* (see syntax diagram) must be specified. The **ypserv** command specifies that **ypxfr** should call back a **yppush** process at the host with IP address *ipadd*, registered as program number *prog*, listening on *port*, and waiting for a response to transaction *tid*.
- d Specifies a *domain* other than the default domain.
- f Forces the transfer to occur even though the version at the server is not more recent than the local version.
- h Gets the map from *host* regardless of the server specified by the map. If *host* is not specified, **ypxfr** asks the YP service for the name of the server, and tries to get the map from there. The *host* parameter may be a name or an IP address of the form *a.b.c.d*.

## Files

/etc/yp/ypxfr.log

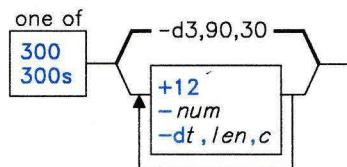
## Related Information

The following commands: "**crontab**" on page 222, "**ypinit**" on page 1243, "**makedbm**" on page 632, "**yppush**" on page 1252, and "**ypserv**" on page 1256.

## Purpose

Handles special line-motion functions for DASI 300/300s work stations.

## Syntax



OL805193

## Description

**Note:** If your work station has a **PLOT** switch, make sure this switch is turned on before using this command.

The **300** command reads standard input, processes its input for printing on the DASI 300, GSI 300, or DTC 300 work stations, and writes to standard output. The **300s** command performs the same functions for the DASI 300s, GSI 300s, and DTC 300s. They convert the input files' motion control characters for half-line forward, half-line reverse, and full-line reverse into motion commands recognized by these work stations.

You can use the **300** and **300s** commands to draw Greek characters and other special symbols that require more than one vertical line, and it allows you to use 12-pitch text. For a discussion of special symbols and Greek characters supported by **300**, see "**greek**" on page 499.

The **nroff** command can be used with the **300** command to format text. **300** must be used if you use special delays or formatting options. You can either pipe from **nroff** to **300** or use the **-T300** flag with **nroff** to specify the printing device. The movement control of the **300** command usually produces better aligned output than **nroff -T300**.

When using **nroff**, the **-s** flag or **.rd** requests are required for inserting paper manually or changing fonts in the middle of a document. In these cases, you must press the line feed key to continue printing.

Using the **300** command with the **neqn** command will give you the best display of your equations. You can use the following sequence to display equations:

```
neqn file . . . | nroff | 300
```

**Note:** Some special characters cannot be correctly printed in column 1 because the print head cannot be moved to the left from that position.

If your output contains Greek characters or reverse line feeds, use a friction-feed platen instead of a forms tractor. A forms tractor slips when reversing direction.

## Flags

**-dt,len,c** Controls output delay factors. The default setting is **-d3,90,30**. DASI 300 is too slow to handle very long lines, too many tab characters, or long strings with no blanks and no identical characters. One null character is inserted in a line for every set of *t* tabs, and for every contiguous string of *c* nonblank, nontab characters. When a line is longer than *len* bytes, several nulls (the line length divided by 20, plus one) are inserted at the end of that line. In all three cases, the nulls delay the output enough to avoid a problem. Items can be omitted from the end of the list, implying the default values. Entering zero for *t* results in insertion of two null bytes per tab, while entering zero for *c* results in insertion of two null bytes per character.

When printing C Language programs, using **-d0,1** will help adjust for the many indentation levels. When printing files like **/etc/passwd**, using **-d3,30,5** will help print it properly.

This flag affects carriage return and line feed delays. The **stty** parameters **n10 cr2** or **n10 cr3** are recommended for most uses.

**-num** Controls the size of half-line spacing. The default half-line values (which are exact half-lines) of *num* are:

10-pitch, 6 lines-per-inch, num = 4  
 12-pitch, 8 lines-per-inch, num = 3  
 12-pitch, 6 lines-per-inch, num = 4

You can use other values for *num* to change the appearance of subscripts and superscripts. For example, **-2** makes **nroff** half-lines act like quarter-lines.

**+12** Uses 12-pitch, 6 lines-per-inch text. The DASI 300 normally allows only two combinations: 10-pitch, 6 lines per inch, or 12-pitch, 8 lines per inch. To use the 12-pitch, 6 lines-per-inch combination, set the PITCH switch to 12 and use the **+12** flag on the command line.

## Related Information

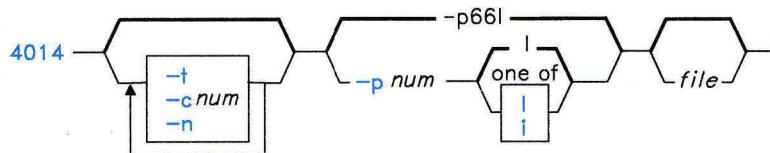
The following commands: “**450**” on page 1265, “**eqn, neqn, checkeq**” on page 395, “**graph**” on page 494, “**mesg**” on page 642, “**nroff, troff**” on page 709, “**stty**” on page 1018, “**tbl**” on page 1053, and “**tplot**” on page 1079.

The **greek** miscellaneous facility in *AIX Operating System Technical Reference*.

## Purpose

Formats a full page 66-line screen display for a Tektronix 4014 work station.

## Syntax



OL805195

## Description

The **4014** command reads a *file* (standard input by default) and writes a 66-line page display to standard output. It also divides the screen into a specified number of columns, adding an eight-space page offset when it uses the default single-column format. It interprets tabs, spaces, backspaces, and TELETYPE Model 37 half-line and reverse-line sequences correctly. At the end of each page, **4014** waits for a line feed from the keyboard before continuing. While **4014** is waiting, you can send commands to the shell by entering *!AIX-cmd*, where *AIX-cmd* is a AIX command.

## Flags

- cnum** Divides the screen into *num* columns and waits after the last column. The default is a single, full page-width column.
- n** Starts displaying at the current cursor position and does not erase the screen.
- pnuml** Sets page length to *num* lines (**l**, the default) or to *num* inches (**i**).
- t** Does not wait between pages.

## Related Information

The following commands: “**pr**” on page 761, “**tc**” on page 1056, and “**troff**” on page 710.

---

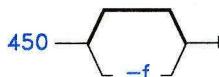
## 450

---

### Purpose

Handles special line-motion functions for the DASI 450 work station

### Syntax



OL805194

### Description

The **450** command reads standard input, processes its data for output on a DASI 450 or an equivalent work station (such as the DIABLO 1620 or XEROX 1700). It converts half-line forward, half-line reverse, and full-line reverse motions to the correct vertical motions on standard output. It attempts to draw Greek characters and other special symbols in the same manner as the **300** command vertical line space. See “**greek**” on page 499 for a list of symbols supported by **450**.

Use **450** with the **nroff -s** flag or **.rd** requests when you need to insert paper manually or change fonts in the middle of a document. Instead of using the **Return** key in these cases, you must use the **LINE FEED** character to get any response. In many cases you can use **nroff -T450** instead of the **450** command. However, you must use **450** if you require special delays or options. In a few cases, using **450** may produce better aligned output. You can pipe the output of the **neqn** command to **450** to print equations neatly.

#### Notes:

1. Make sure the **PLOT** switch is turned on before using this command. Also, the **SPACING** switch should be in the desired position, either 10- or 12-pitch. For either setting, vertical spacing is 6 lines per inch unless changed to 8 lines per inch by an escape sequence.
2. Some special characters cannot be correctly printed in column 1 because the print head cannot be moved to the left from that position.
3. If your output contains Greek characters or reverse line feeds, use a friction-feed platen instead of a forms tractor. A forms tractor tends to slip when reversing direction.

## Flag

- f Permits the use of ETX/ACK protocol with 1200 bps printers. You cannot use **450** with this flag in a pipeline or if you redirect its output. Instead it must drive the printer directly.

## Related Information

The following commands: “**300**” on page 1262, “**eqn, neqn, checkeq**” on page 395, “**graph**” on page 494, “**greek**” on page 499, “**mesg**” on page 642, “**nroff, troff**” on page 709, “**stty**” on page 1018, “**tabs**” on page 1041, “**tbl**” on page 1053, “**tplot**” on page 1079, and “**troff**” on page 710.

The **greek** miscellaneous facility in *AIX Operating System Technical Reference*.

---

## Appendix A. AIX Device Table

AIX standard devices are special files. The following table lists and describes some special files. For more detailed information on special files, see *AIX Operating System Technical Reference*.

<b>Special File</b>	<b>Description</b>
<b>appltrace</b>	Application trace pseudo-device driver
<b>config</b>	Configuration pseudo-device driver
<b>console</b>	Console device
<b>error</b>	Error-logging interface
<b>fd[num]</b>	Diskette drive; block device
<b>fp</b>	Floating-point function
<b>hd[num]</b>	Fixed disk drive; block device
<b>hft</b>	High function terminal
<b>kmem</b>	Kernel memory image
<b>lp[num]</b>	Line printer
<b>mem</b>	Memory image
<b>null</b>	The null device
<b>nvrnm</b>	Non-volatile memory image
<b>osm</b>	System message interface
<b>prf</b>	AIX Operating System profiler
<b>rfd[num]</b>	Diskette drive; raw device
<b>rhd[num]</b>	Fixed disk drive; raw device
<b>rmt[num]</b>	Streaming tape
<b>termio</b>	General terminal interface
<b>tty[num]</b>	Controlling terminal interface
<b>unixtrace</b>	Kernel trace event pseudo-device driver
<b>vrntrace</b>	VRM trace event pseudo-device driver

Figure 13. AIX Standard Devices (Special Files)

---

## Appendix B. Program Cross-Reference

Command	Program	Command	Program
<b>acct/*</b>	Multi-User Services	<b>auditwrite</b>	AIX Operating System
<b>chargefee</b>		<b>awk</b>	AIX Operating System
<b>ckpacct</b>		<b>back</b>	Extended Services
<b>dosdisk</b>		<b>backup</b>	AIX Installation/Maintenance
<b>lastlogin</b>		<b>banner</b>	Extended Services
<b>monacct</b>		<b>basename,</b>	AIX Operating System
<b>nulladm</b>		<b>dirname</b>	
<b>prctmp</b>		<b>bc</b>	Extended Services
<b>prdaily</b>		<b>bdiff</b>	Extended Services
<b>prtacct</b>		<b>bellmail</b>	AIX Operating System
<b>remove</b>		<b>bffcreate</b>	AIX Operating System
<b>shutacct</b>		<b>bfs</b>	Extended Services
<b>startup</b>		<b>biodd_cfg</b>	Extended Services
<b>turnacct</b>		<b>bj</b>	Extended Services
<b>acctcms</b>	Multi-User Services	<b>bs</b>	Extended Services
<b>acctcom</b>	Multi-User Services	<b>burst</b>	Extended Services
<b>acctcon</b>	Multi-User Services	<b>cal</b>	Extended Services
<b>acctcon1</b>		<b>calendar</b>	Extended Services
<b>acctcon2</b>		<b>cat</b>	AIX Operating System
<b>acctdisk,</b>	Multi-User Services	<b>cb</b>	AIX Operating System
<b>acctdusg</b>		<b>cc</b>	AIX Operating System
<b>acctmerg</b>	Multi-User Services	<b>cd</b>	AIX Operating System
<b>acctprc</b>	Multi-User Services	<b>cdc</b>	Extended Services
<b>acctprc1</b>		<b>cflow</b>	Extended Services
<b>acctprc2</b>		<b>chgrp</b>	AIX Operating System
<b>accton</b>		<b>chkcomp</b>	AIX Operating System
<b>actman</b>	AIX Operating System	<b>chmod</b>	AIX Operating System
<b>adb</b>	Extended Services	<b>chgstate</b>	AIX Operating System
<b>admin</b>	Extended Services	<b>installc</b>	
<b>ali</b>	Extended Services	<b>updatec</b>	
<b>anno</b>	Extended Services	<b>chown</b>	AIX Operating System
<b>ap</b>	Extended Services	<b>chparm</b>	AIX Installation/Maintenance
<b>ar</b>	AIX Operating System	<b>chroot</b>	Extended Services
<b>arithmetic</b>	Extended Services	<b>chtcb</b>	AIX Operating System
<b>as</b>	AIX Operating System	<b>clri</b>	AIX Operating System
<b>at, batch</b>	AIX Operating System	<b>cmp</b>	AIX Operating System
<b>audit</b>	AIX Operating System	<b>col</b>	Extended Services
<b>auditapp</b>	AIX Operating System	<b>comb</b>	Extended Services
<b>auditbin</b>	AIX Operating System	<b>comm</b>	Extended Services
<b>auditpr</b>	AIX Operating System	<b>comp</b>	Extended Services
<b>auditselect</b>	AIX Operating System	<b>confer</b>	Multi-User Services
<b>auditstream</b>	AIX Operating System	<b>config</b>	AIX Operating System

Command	Program	Command	Program
<b>conflict</b>	Extended Services	<b>dosread</b>	AIX Operating System
<b>connect</b>	AIX Operating System	<b>doswrite</b>	AIX Operating System
<b>cp</b>	AIX Operating System	<b>dp</b>	Extended Services
<b>cpio</b>	AIX Operating System	<b>dsipc</b>	AIX Operating System
<b>cpp</b>	AIX Operating System	<b>dsldxprof</b>	Distributed Services
<b>craps</b>	Extended Services	<b>dsstate</b>	Distributed Services
<b>crash</b>	AIX Operating System	<b>dsxlate</b>	Distributed Services
<b>cron</b>	AIX Operating System	<b>du</b>	AIX Operating System
<b>crontab</b>	AIX Operating System	<b>dump</b>	Extended Services
<b>csch</b>	AIX Operating System	<b>dumpfmt</b>	AIX Operating System
<b>csplit</b>	Extended Services	<b>echo</b>	AIX Operating System
<b>ct</b>	Extended Services	<b>ed</b>	AIX Installation/Maintenance
<b>ctab</b>	AIX Operating System	<b>edconfig</b>	Extended Services
<b>ctags</b>	Extended Services	<b>edit</b>	Extended Services
<b>cu</b>	Extended Services	<b>env</b>	AIX Operating System
<b>cut</b>	AIX Operating System	<b>eqn, neqn,</b>	Extended Services
<b>cvid</b>	AIX Operating System	<b>checkeq</b>	
<b>Cvt</b>	Extended Services	<b>errdead</b>	AIX Operating System
<b>cw, checkcw</b>	Extended Services	<b>errdemon</b>	AIX Operating System
<b>cxref</b>	Extended Services	<b>errpt, errpd</b>	AIX Operating System
<b>date</b>	AIX Operating System	<b>errstop</b>	AIX Operating System
<b>dbx</b>	Extended Services	<b>errupdate</b>	AIX Operating System
<b>dc</b>	Extended Services	<b>ex</b>	Extended Services
<b>dcopy</b>	Extended Services	<b>expr</b>	AIX Operating System
<b>dd</b>	AIX Installation/Maintenance	<b>factor</b>	Extended Services
<b>defkey</b>	AIX Operating System	<b>ff</b>	Extended Services
<b>del</b>	AIX Operating System	<b>file</b>	AIX Operating System
<b>delta</b>	Extended Services	<b>find</b>	AIX Operating System
<b>deroff</b>	Extended Services	<b>fish</b>	Extended Services
<b>devices</b>	AIX Operating System	<b>fmt</b>	AIX Operating System
<b>devnm</b>	AIX Operating System	<b>folder</b>	Extended Services
<b>df</b>	AIX Operating System	<b>folders</b>	Extended Services
<b>diff</b>	AIX Operating System	<b>format</b>	AIX Installation/Maintenance
<b>diff3</b>	Extended Services	<b>fortune</b>	Extended Services
<b>diffmk</b>	Extended Services	<b>forw</b>	Extended Services
<b>dircmp</b>	Extended Services	<b>fptype</b>	AIX Operating System
<b>diskusg</b>	Multi-User Services	<b>fsck, dfsck</b>	AIX Installation/Maintenance
<b>display</b>	AIX Operating System	<b>fsdb</b>	AIX Installation/Maintenance
<b>dist</b>	Extended Services	<b>fuser</b>	Extended Services
<b>dos</b>	Extended Services	<b>fwtmp</b>	Multi-User Services
<b>dosdel</b>	AIX Operating System	<b>acctwtmp</b>	
<b>dosdir</b>	AIX Operating System	<b>wtmpfix</b>	

Command	Program	Command	Program
<b>gdev</b>	Multi-User Services	<b>mvmd</b>	
<b>hpd</b>		<b>ipcrm</b>	AIX Operating System
<b>erase</b>		<b>ipcs</b>	AIX Operating System
<b>hardcopy</b>		<b>iptable</b>	Distributed Services
<b>tekset</b>		<b>istat</b>	Extended Services
<b>td</b>		<b>join</b>	Extended Services
<b>ged</b>	Multi-User Services	<b>keyboard</b>	AIX Operating System
<b>gend</b>	Multi-User Services	<b>kill</b>	AIX Operating System
<b>get</b>	Extended Services	<b>killall</b>	AIX Operating System
<b>getopt</b>	AIX Operating System	<b>ld</b>	AIX Operating System
<b>gettext</b>	AIX Operating System	<b>lex</b>	Extended Services
<b>getty</b>	AIX Operating System	<b>li</b>	AIX Operating System
<b>graph</b>	Multi-User Services	<b>line</b>	AIX Operating System
<b>graphics</b>	Multi-User Services	<b>link, unlink</b>	AIX Operating System
<b>greek</b>	Multi-User Services	<b>lint</b>	Extended Services
<b>grep</b>	AIX Operating System	<b>ln</b>	AIX Operating System
<b>groups</b>	AIX Operating System	<b>locator</b>	AIX Operating System
<b>gutil</b>	Extended Services	<b>login</b>	AIX Operating System
<b>bel</b>		<b>loginx</b>	AIX Operating System
<b>cvrtopt</b>		<b>logname</b>	AIX Operating System
<b>gd</b>		<b>logout</b>	AIX Operating System
<b>gtop</b>		<b>lorder</b>	AIX Operating System
<b>pd</b>		<b>lp</b>	AIX Operating System
<b>ptog</b>		<b>ls</b>	AIX Installation/Maintenance
<b>quit</b>		<b>m4</b>	AIX Operating System
<b>remcom</b>		<b>mail, Mail</b>	AIX Operating System
<b>whatis</b>		<b>mailstats</b>	Extended Services
<b>yoo</b>		<b>make</b>	AIX Operating System
<b>hangman</b>	Extended Services	<b>makekey</b>	AIX Operating System
<b>help</b>	Extended Services	<b>man</b>	AIX Operating System
<b>hp</b>	Multi-User Services	<b>mark</b>	Extended Services
<b>hyphen</b>	Extended Services	<b>mdrc</b>	AIX Operating System
<b>id</b>	Multi-User Services	<b>mesg</b>	AIX Operating System
<b>inc</b>	Extended Services	<b>mhl</b>	Extended Services
<b>init</b>	AIX Installation/Maintenance	<b>mhmail</b>	Extended Services
<b>install</b>	AIX Operating System	<b>mhpath</b>	Extended Services
<b>install-mh</b>	Extended Services	<b>minidisks</b>	AIX Operating System
<b>installp</b>	AIX Operating System	<b>mkdir</b>	AIX Operating System
<b>inusave</b>		<b>mkfs</b>	AIX Installation/Maintenance
<b>inurecv</b>		<b>mknod</b>	AIX Installation/Maintenance
<b>inurest</b>		<b>mm, checkmm</b>	Extended Services
<b>ckprereq</b>		<b>mmt, checkmm</b>	Extended Services

Command	Program	Command	Program
<b>moo</b>	Extended Services	<b>proto</b>	Extended Services
<b>mount</b>	AIX Installation/Maintenance	<b>prs</b>	Extended Services
<b>msgchk</b>	Extended Services	<b>ps</b>	AIX Operating System
<b>msh</b>	Extended Services	<b>pstart, penable, pshare, pdelay</b>	AIX Operating System
<b>mv</b>	AIX Operating System	<b>ptx</b>	Extended Services
<b>mmdir</b>	AIX Operating System	<b>puttext</b>	AIX Operating System
<b>ncheck</b>	AIX Operating System	<b>pwck</b>	Extended Services
<b>ndtable</b>	Distributed Services	<b>pwd</b>	AIX Operating System
<b>newform</b>	Extended Services	<b>pwtable</b>	Distributed Services
<b>newgrp</b>	AIX Operating System	<b>qdaemon</b>	AIX Operating System
<b>news</b>	AIX Operating System	<b>quiz</b>	Extended Services
<b>next</b>	Extended Services	<b>rc</b>	AIX Operating System
<b>nice</b>	AIX Operating System	<b>rcvdist</b>	Extended Services
<b>nl</b>	Extended Services	<b>rcvpack</b>	Extended Services
<b>nm</b>	AIX Operating System	<b>rcvstore</b>	Extended Services
<b>nohup</b>	AIX Operating System	<b>rcvtty</b>	Extended Services
<b>nroff, troff</b>	Extended Services	<b>refile</b>	Extended Services
<b>number</b>	Extended Services	<b>regcmp</b>	Extended Services
<b>od</b>	Extended Services	<b>repl</b>	Extended Services
<b>open</b>	AIX Operating System	<b>restore</b>	AIX Installation/Maintenance
<b>pack</b>	Extended Services	<b>rm</b>	AIX Installation/Maintenance
<b>pcat</b>		<b>rmail</b>	AIX Operating System
<b>unpack</b>		<b>rm del</b>	Extended Services
<b>packf</b>	Extended Services	<b>rmdir</b>	AIX Installation/Maintenance
<b>passwd</b>	AIX Operating System	<b>rmf</b>	Extended Services
<b>paste</b>	AIX Operating System	<b>rmm</b>	Extended Services
<b>pdisable, phold</b>	AIX Operating System	<b>rstatd</b>	AIX Operating System
<b>pg</b>	AIX Operating System	<b>runacct</b>	Multi-User Services
<b>pick</b>	Extended Services	<b>rusersd</b>	AIX Operating System
<b>piobe</b>	AIX Operating System	<b>rwalld</b>	AIX Operating System
<b>portmap</b>	AIX Operating System	<b>sact</b>	Extended Services
<b>post</b>	Extended Services	<b>sadc</b>	Multi-User Services
<b>pr</b>	AIX Operating System	<b>sa1</b>	
<b>prev</b>	Extended Services	<b>sa2</b>	
<b>print</b>	AIX Operating System	<b>sag</b>	Multi-User Services
<b>prof</b>	Extended Services	<b>sar</b>	Multi-User Services
<b>profiler</b>	Extended Services	<b>scan</b>	Extended Services
<b>prfld</b>		<b>sccsdiff</b>	Extended Services
<b>prfstat</b>		<b>sdb</b>	Extended Services
<b>prfdc, prfsnap</b>		<b>sdiff</b>	Extended Services
<b>prfpr</b>		<b>secure</b>	AIX Operating System
<b>prompter</b>	Extended Services		

Command	Program	Command	Program
<b>sed</b>	AIX Operating System	<b>test</b>	AIX Operating System
<b>send</b>	Extended Services	<b>tic</b>	Extended Services
<b>sendmail</b>	AIX Operating System	<b>time</b>	AIX Operating System
<b>setdma</b>	AIX Operating System	<b>timex</b>	Multi-User Services
<b>setmnt</b>	AIX Operating System	<b>tlog</b>	AIX Operating System
<b>sh</b>	AIX Installation/Maintenance	<b>tlogger</b>	AIX Operating System
<b>shell</b>	AIX Operating System	<b>toc</b>	Multi-User Services
<b>shlib</b>	AIX Operating System	<b>dtoc</b>	
<b>show</b>	Extended Services	<b>ttoc</b>	
<b>shutdown</b>	AIX Operating System	<b>vtoc</b>	
<b>size</b>	AIX Operating System	<b>touch</b>	AIX Operating System
<b>skulker</b>	AIX Operating System	<b>tplot</b>	Multi-User Services
<b>sleep</b>	AIX Operating System	<b>tput</b>	Extended Services
<b>slocal</b>	Extended Services	<b>tr</b>	Extended Services
<b>sno</b>	Extended Services	<b>trace</b>	AIX Operating System
<b>sort</b>	AIX Operating System	<b>trcrpt</b>	AIX Operating System
<b>sortm</b>	Extended Services	<b>trcstop</b>	AIX Operating System
<b>sound</b>	AIX Operating System	<b>trcupdate</b>	AIX Operating System
<b>sprayd</b>	AIX Operating System	<b>trdiag</b>	Extended Services
<b>spell</b>	Extended Services	<b>true</b>	AIX Operating System
<b>spline</b>	Multi-User Services	<b>tsh</b>	AIX Operating System
<b>split</b>	AIX Operating System	<b>tsort</b>	AIX Operating System
<b>splp</b>	AIX Operating System	<b>ttt</b>	Extended Services
<b>spost</b>	Extended Services	<b>tty</b>	AIX Operating System
<b>stat</b>	Multi-User Services	<b>turnon</b>	Extended Services
<b>strip</b>	AIX Operating System	<b>tvi</b>	AIX Operating System
<b>stty</b>	AIX Installation/Maintenance	<b>ugtable</b>	Distributed Services
<b>su</b>	AIX Operating System	<b>umask</b>	AIX Operating System
<b>sum</b>	AIX Operating System	<b>umount,</b>	AIX Installation/Maintenance
<b>sync</b>	AIX Installation/Maintenance	<b>unmount</b>	
<b>sysck</b>	AIX Operating System	<b>uname</b>	AIX Operating System
<b>syslogd</b>	Multi-User Services	<b>unget</b>	Extended Services
<b>tab, untab</b>	Extended Services	<b>uniq</b>	AIX Operating System
<b>tabs</b>	Extended Services	<b>units</b>	Extended Services
<b>tail</b>	Extended Services	<b>updatep</b>	AIX Operating System
<b>tapechk</b>	AIX Operating System	<b>inudocm</b>	
<b>tar</b>	Extended Services	<b>inuupdt</b>	
<b>tbl</b>	Extended Services	<b>users, adduser</b>	AIX Operating System
<b>tc</b>	Multi-User Services	<b>uucheck</b>	Extended Services
<b>tctl</b>	AIX Installation/Maintenance	<b>uucico</b>	Extended Services
<b>tee</b>	AIX Operating System	<b>uucleanup</b>	Extended Services
<b>termdef</b>	AIX Operating System	<b>uucp</b>	Extended Services

---

<b>Command</b>	<b>Program</b>
<b>uulog</b>	Extended Services
<b>uuname</b>	Extended Services
<b>uupick</b>	Extended Services
<b>uusched</b>	Extended Services
<b>uustat</b>	Extended Services
<b>uuto</b>	Extended Services
<b>uutry, Uutry, uukick</b>	Extended Services
<b>uux</b>	Extended Services
<b>uuxqt</b>	Extended Services
<b>val</b>	Extended Services
<b>varyoff</b>	AIX Operating System
<b>varyon</b>	AIX Operating System
<b>vc</b>	Extended Services
<b>verify</b>	AIX Operating System
<b>vi, vedit, view</b>	Extended Services
<b>vmh</b>	Extended Services
<b>vrm2rtfont</b>	AIX Operating System
<b>vrmconfig</b>	AIX Installation/Maintenance
<b>wall</b>	AIX Operating System
<b>watch</b>	AIX Operating System
<b>wc</b>	AIX Operating System
<b>what</b>	AIX Operating System
<b>whatnow</b>	Extended Services
<b>who</b>	AIX Operating System
<b>whom</b>	Extended Services
<b>write</b>	AIX Operating System
<b>writesrv</b>	AIX Operating System
<b>wump</b>	Extended Services
<b>xargs</b>	AIX Operating System
<b>xdbx</b>	Extended Services
<b>yacc</b>	Extended Services
<b>300</b>	Multi-User Services
<b>4014</b>	Multi-User Services
<b>450</b>	Multi-User Services



---

## Appendix C. Syntax Diagram Guide

---

## Syntax Diagrams

Syntax diagrams are read from left to right. A single path is followed until the end mark is encountered. An end mark indicates nothing more can be entered with that command.

Within syntax diagrams, the following conventions are used:

- Diagram items that must be entered literally on the command line are in **bold**. These items include the command name, flags, and literal characters.
- Variable diagram items for which you determine the name to enter are in *italics*. These items include parameters that follow flags and parameters that the command reads, such as *files* and *directories*.
- Default values that do not have to be entered are in the normal font on a **bold** path.

## Command Only

The simplest syntax diagram illustrates a command that is entered on the command line with nothing else. For example:

```
logname —|
```

OL805145

The bold command name means **logname** should be entered literally. As you follow the line away from the command name to the right, you reach the end mark and must stop. To enter this command, you would enter:

```
logname
```

## Commands with Required Parameters

Many diagrams have parameters to represent specific values that you must enter on the command line. When you encounter a parameter, read the command description to determine what to enter in place of the parameter. The following diagram requires a parameter:

```
unlink — file —|
```

OL805227

The bold command name means **unlink** should be entered literally. As you follow the line away from the command name to the right, the next item that you encounter is the parameter *file*. Since the only path goes through this parameter, you must supply a file name. As you move further to the right, you reach the end mark and must stop. To enter this command, enter:

```
unlink report
```

---

Suppose you want to unlink two files, **report** and **memo**. Since this command accepts one and only one file, you would have to enter the command twice:

```
unlink report
unlink memo
```

Some commands allow you to enter more than one parameter on the command line. If you are allowed to do so, the diagrams show it with a **repeat arrow**, an arrow that provides a path back to an earlier part of the diagram. For example:

```
sact 
```

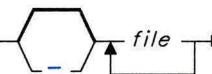
OL805063

The bold command name indicates **sact** should be entered literally. As you follow the line away from the command name to the right, the next item that you encounter is the parameter *file*. Since the only path goes through this parameter, you must enter a file name. As you move further to the right, you reach a repeat arrow. Here you can choose to remain on the main path and proceed to the end mark or follow the repeat arrow around to the point between the command name and the parameter. Following the repeat arrow allows you to select the parameter again. If there is a maximum number of parameters that you can enter, the diagram tells you that number. If no maximum is specified (as in this diagram), then you can choose the repeat arrow again and again until you reach the limit of the length of a command line. Here are some examples:

```
sact s.letter
sact s.letter s.memo s.report
```

## Commands with an Optional Flag or Parameter

Many commands have optional flags or parameters. If something is optional, you have a choice of paths in the diagram. One takes you around the item, and the other takes you through it. For example:

```
del 
```

OL805049

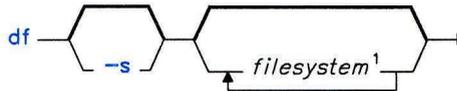
With this diagram, as you move to the right from the command name, you reach a branch. You can either take the upper branch (nothing is entered) or you can take the lower branch (enter the flag). The bold line above the flag is the default line. This is the most commonly used path. After the branch, you encounter a parameter. Since the only path goes through the parameter you must enter it. After the parameter is entered once you can choose to proceed to the end mark or use the repeat arrow to select the parameter again.

---

For example:

```
del file1
del file1 file2 file3
del - file1 file2
```

As the command syntax becomes more complicated, the features of the diagrams are combined to help you enter commands properly. The next diagram shows a command that accepts an optional flag and an optional parameter that can be repeated.



<sup>1</sup> The default action is to provide information for each file system in `/etc/filesystems` with the attribute `free=true`.

OL805052

In this diagram, as you move to the right from the command name, you reach a branch. You can either take the upper branch (nothing is entered) or the lower branch (enter the flag). Next you encounter another branch. Here you can either take the upper branch (nothing is entered) or the lower branch (enter the parameter). If you choose to enter the parameter, you can enter the parameter once and proceed to the end mark or you can use the repeat arrow to select the parameter again. For example:

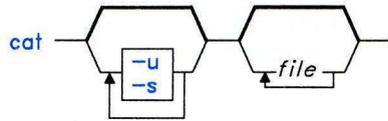
```
df
df -s
df system1
df -s system1
df system1 system2
df -s system1 system2 system3
```

## Commands That Take More Than One Flag

With many commands, you can enter as many items from a group of flags or parameters as you want within the limits of the length of the command line. If this is the case, the items are in a box that has a repeat arrow around it. Follow the arrow around and through the box as many times as necessary to select all of the items you want to use. Note that most commands do not work properly if you choose the same flags more than once. Therefore, once you have chosen an item from the box, you should not choose it again unless a footnote indicates a flag may be used more than once.

---

For example:



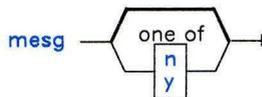
OL805086

With this command you can enter only the command name by following the default line over the box, enter one flag and then continue to the end mark, or follow the arrow around and choose both flags. The following are examples of valid command lines:

```
cat
cat -u
cat -s
cat -u -s
```

## Commands with an Exclusive Flag or Parameter

Many commands have flags or parameters that should not be entered together on the command line. Mutually exclusive items are enclosed in a single-choice box (a box with the words *one of* above it). You can choose only one item from this type of box. The following diagram contains a single-choice box.



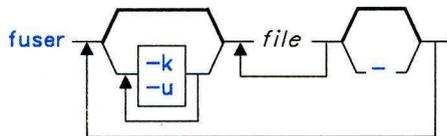
OL805036

Valid ways to enter this command are:

```
msg
msg n
msg y
```

## Commands that Can Repeat Part of a Sequence

Some commands allow you to choose flags for each parameter that they read. When this is the case, more than one repeat arrow allows you to go back to earlier parts of the diagram. For example:



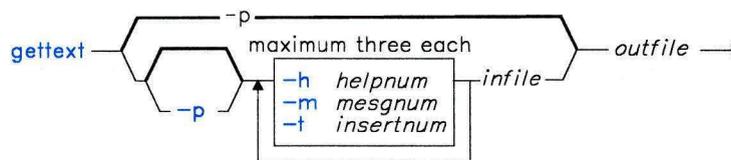
In this diagram, there are three repeat arrows. The first allows you to choose one or both flags. The second allows you to have **fuser** read more than one file. The third allows you to repeat the complete sequence from the beginning of the diagram to the end. The following are all correct ways to enter **fuser**:

```
fuser memo
fuser memo -
fuser -k memo
fuser -k -u memo
fuser -k -u memo letter
fuser -k -u memo -
fuser -k memo - -u -k letter -u report -
```

The third arrow allows you to enter the same flag repeatedly, but only *after* at least one file name has been entered. If you follow the diagram, you cannot repeat a flag without entering at least one file name after it.

## Commands with Default Values

The default line can show more than just an alternate path around flags and parameters. Sometimes, a flag is set automatically (by default) or a sometimes a parameter has a default value. When this is the case, the default value is shown in the normal font on the default line. For example:



OL805130

If you do not enter any flags with **gettext**, the **-p** flag is set by default. If you choose the path that contains the **-h**, **-m**, and **-t** flags, you must choose whether or not to use the **-p** flag also. The following command lines are equivalent ways of entering **gettext**:

```
gettext -p report
gettext report
```

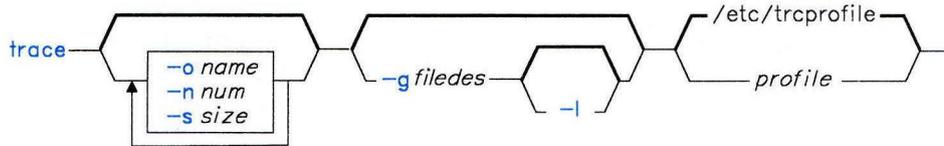
The following are valid command lines that do not use the default flag for **gettext**:

```
gettext -h2 report
gettext -m3 memo report
```

To select both the **-p** and **-m** flag, you must explicitly enter the **-p** flag. For example:

```
gettext -p -m3 memo
```

You can also have default parameter values.



OL805279

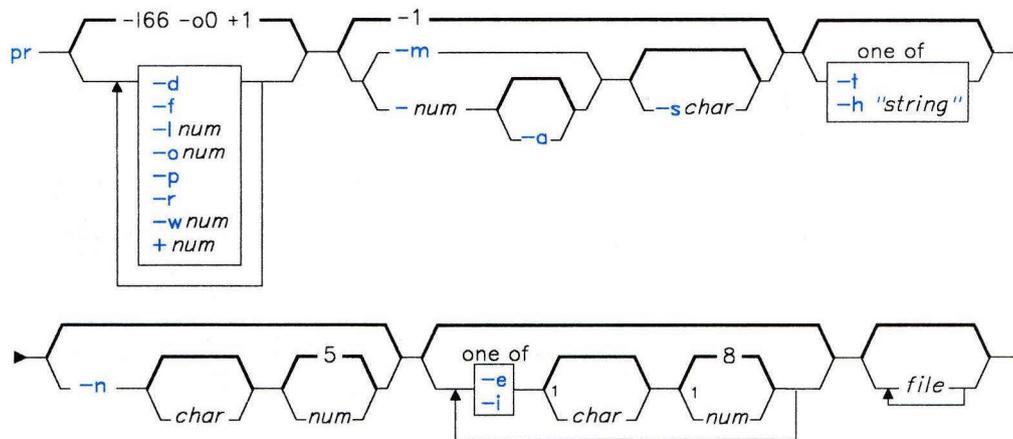
In this case you can choose to specify a *profile* or to use the default value `/etc/trcprofile`. Since the bold path indicates that this is a default, you do not have to enter this file name. The system does this for you. If a *profile* is not supplied, `trace` reads the file `/etc/trcprofile`.

The following are equivalent command lines:

```
trace
trace /etc/trcprofile
```

## Diagrams That Are Continued on the Next Line

Some of the more complex diagrams cannot fit on one line. They are marked with an arrowhead where they break, and they continue with the arrowhead on the next line. For example:



<sup>1</sup> Do not put a blank between these items.

OL805437

Follow the diagram making choices until you reach the arrowhead. Then go down to the arrowhead on the next line. Continue until you reach the end mark. Following the diagram will seem to impose a specific order to the flags; however, you do not need to strictly follow that order when entering the command. If strict order is important, it is stated under "Description" in the commands discussion.

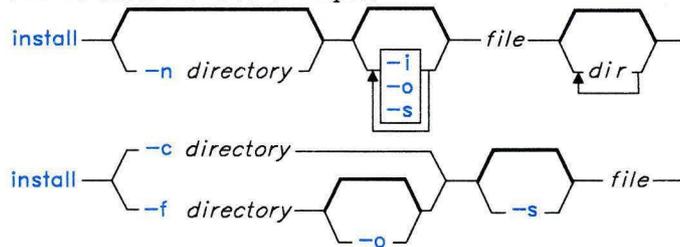
The following are some of the ways you can enter **pr** on the command line:

```
pr
pr -d
pr -o4 -r -m -sX memo letter
pr -r -m -t -n4 -iX3 memo letter report
pr -m -n4 -r -iX3 -t memo report letter
pr -l30 5 -3 -a -nX -iX3 -eY memo report
```

Notice that this diagram has a footnote. Footnotes are used to show information that cannot be diagrammed. In this case, it tells you that you cannot put a space between the **-e** or **-i** flags and their parameters.

## Commands With More Than One Diagram

Some commands require more than one diagram to indicate the different ways a command can be entered. For example:



OL805022

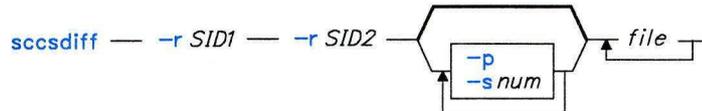
Because some flags and parameters cannot be combined with others, two diagrams are required to indicate the ways the command can be entered. For example, the following are ways you can enter this command:

```
install -o fixit /etc /games
install -n /usr/bin fixit
install -c /usr/bin fixit
install -f /usr/bin -o -s fixit
```

---

## File Input and Output in Diagrams

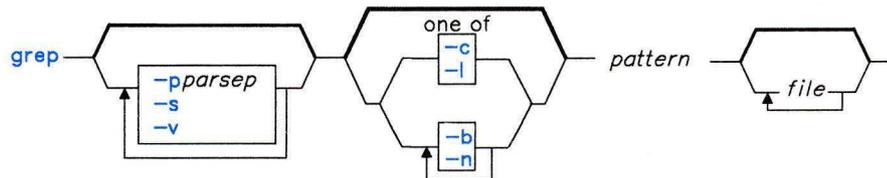
Some commands must read a file as their input, some must read standard input, and some can read both. The syntax diagrams help you to determine which case applies to each particular command. If a command must read a file as its input, the diagram shows a path through a parameter representing the file and the “Description” section tells you this file is an input file. The path in the diagram will not have a branch around it. The following is an example of a command that must read an input file:



OL805258

When there is no place in a diagram to supply an input file, the command reads standard input for this information. To supply input through standard input, you can supply it through a pipeline, through redirection of the output from another command, or directly from the keyboard (if it is standard input).

Most commands can either read standard input or files for their input. The diagrams show this by branching and giving you a choice of entering a file name or nothing. For example:



OL805375

If you specify an input file, the command reads that file for its input. If you do not specify a file, the command reads standard input. The following are valid command lines for this command:

```
grep AAA
grep -sAAA
grep AAA memo
grep -s -v AAA memo
grep AAA memo letter report
```

In the first two cases, the **grep** command reads standard input. In the last three cases, the **grep** command uses the specified input files.

**Note:** Some commands require that you enter a - (minus) when you want the command to read standard input. If this is the case, it is explained under “Description”, not in the diagram. Usually this is done so that you can read several files as input and you can include standard input as one of the files.



---

## Appendix D. Japanese Language Support

The following table lists the commands that have not been modified to support Japanese characters.

acct/*	dist	ipcrm
acctcms	dos	ipcs
acctcom	dosdel	ipctable
acctcon1, acctcon2	dp	istat
acctdisk, acctdusg	dsipc	keyboard
acctmerg	dsldxprof	lint
acctprc1, acctprc2, accton	dsstate	locator
acctman	dsxlate	m4
adb	dumpfmt	mail, Mail
ali	edconfig	mailstats
anno	edit	makekey
ap	eqn, neqn, checkeq	man
arithmetic	errdead	mark
as	errdemon	mdrc
back	errpt, errpd	mhl
bellmail	errstop	mhmail
bffcreate	errupdate	mhpath
bj	ex	mm, checkmm
burst	factor	mmt, mant, mmt, mvt, checkmm
cc, fcc, vcc	fish	moo
cd	fmt	msgchk
cflow	folder	msh
chkcomp	folders	ncheck
chgstate	fortune	ndtable
col	forw	newgrp
comp	fptype	next
confer, joinconf	fuser	nroff, troff
conflict	fwtmp, acctwtmp, wtmpfix	number
connect	gdev - hpd, erase, hardcopy, tekset, td	open
cpp	ged	packf
craps	gend	pick
ct	gettext	post
ctags	graphics	prev
cu	greek	prof
cvid	groups	profiler - prfld, prfstat, prfdc, prfsnap, prfpr
Cvt	gutil - bel, cvrtopt, gd, gtop, pd, ptog, quit, remcom, whatis, yoo	prompter
cw, checkcw	hangman	proto
cxref	hp	pstart, pshare, pdelay
dbx	hyphen	ptx
deroff	inc	puttext
diffmk	install	pwtable
diskusg	install-mh	quiz
display	installp	rc

---

rcvdist	stat	uucico
rcvpack	syslogd	uucleanup
rcvstore	tapechk	uulog
rcvtty	tbl	uuname
refile	tc	uupick
repl	termdef	uusched
rmail	test	uustat
rmf	tic	uuto
rmm	tlog	uutry, Uutry, uukick
runacct	tlogger	uux
sact	toc - dtoc, ttoc, vtoc	uuxqt
scan	tplot	varyoff
send	trace	varyon
sendmail, mailq, newaliases	trcrpt	verify
setdma	trcstop	vmh
show	trcupdate	vrn2rtfont
slocal	trdiag	vrnconfig
sno	ttt	whatnow
sortm	turnon, turnoff	whom
sound	ugtable	writesrv
spell, hashmake, spellin, hashcheck	umask	wump
splp	units	xdbx
spost	updatep, inudocm, inuupdt	300, 300s
	uucheck	4014
		450



---

## Glossary

**access.** To obtain data from or put data in storage.

**access permission.** A group of designations that determine who can access a particular AIX file and how the user may access the file.

**account.** The login directory and other information that give a user access to the system.

**active service.** A code service relationship between a client and a server where the client is dependent on accessing overmounted server directories to get execution access to required programs and functions. Contrast with *passive service*. See also *client* and *server*.

**activity manager.** A collection of system-supplied tasks allowing users to manage their activities. Provides the ability to list current activities (Activity List) and to begin, cancel, hide, and activate activities.

**All Points Addressable (APA) display.** A display that allows each pel to be individually addressed. An APA display allows for images to be displayed that are not made up of images predefined in character boxes. Contrast with *character display*.

**allocate.** To assign a resource, such as a disk file or a diskette file, to perform a specific task.

**alphabetic.** Pertaining to a set of letters a through z.

**alphanumeric character.** Consisting of letters, numbers and often other symbols, such as punctuation marks and mathematical symbols.

**American National Standard Code for Information Interchange (ASCII).** The code developed by ANSI for information interchange among data processing systems, data communications systems, and associated equipment. The ASCII character set consists of 7-bit control characters and symbolic characters.

**American National Standards Institute (ANSI).** An organization sponsored by the Computer and Business Equipment Manufacturers Association for establishing voluntary industry standards.

**application.** A program or group of programs that apply to a particular business area, such as the Inventory Control or the Accounts Receivable application.

**application program.** A program used to perform an application or part of an application.

**argument.** Numbers, letters, or words that change the way a command works.

**ASCII.** See *American National Standard Code for Information Interchange*.

**attribute.** A characteristic. For example, the attribute for a displayed field could be blinking.

**audit.** To review and examine the activities of a data processing system mainly to test the adequacy and effectiveness of procedures for data privacy and data integrity.

**audit bin.** A file containing unprocessed audit records.

**audit class.** A list of events that define which actions taken on a system are recorded. They

---

are defined by the administrator of the system in the user data base.

**audit event.** An action (such as a command or access) taken on a system, which can be recorded by the system.

**audit trail.** A collection of events that could compromise system security recorded in the order in which they occurred.

**auto carrier return.** The system function that places carrier returns automatically within the text and on the display. This is accomplished by moving whole words that exceed the line end zone to the next line.

**backend.** The program that sends output to a particular device. There are two types of backends: friendly and unfriendly.

**background process.** (1) A process that does not require operator intervention that can be run by the computer while the work station is used to do other work. (2) A mode of program execution in which the shell does not wait for program completion before prompting the user for another command.

**backup copy.** A copy, usually of a file or group of files, that is kept in case the original file or files are unintentionally changed or destroyed.

**backup diskette.** A diskette containing information copied from a fixed disk or from another diskette. It is used in case the original information becomes unusable.

**backup format.** A compressed file format. When the backup command makes a copy of a file, it writes the file in this format. A file in this format must be restored by the restore command before it can be used.

**backup format file.** (1) A file in backup format. (2) In a code-server environment, a file in backup format that contains a copy of an install or update distribution media for a program.

**bad block.** A portion of a disk that can never be used reliably.

**base address.** The beginning address for resolving symbolic references to locations in storage.

**base name.** The last element to the right of a full path name. A filename specified without its parent directories.

**batch printing.** Queueing one or more documents to print as a separate job. The operator can type or revise additional documents at the same time. This is a background process.

**batch processing.** A processing method in which a program or programs process records with little or no operator action. This is a background process. Contrast with *interactive processing*.

**big word.** A collection of alphanumeric characters defined by the collation table and bounded by blanks, tabs, or new-line indicators.

**binary.** (1) Pertaining to a system of numbers to the base two; the binary digits are 0 and 1. (2) Involving a choice of two conditions, such as on-off or yes-no.

**bit.** Either of the binary digits 0 or 1 used in computers to store information. See also *byte*.

**block.** (1) A group of records that is recorded or processed as a unit. Same as *physical record*. (2) In data communications, a group of records that is recorded, processed, or sent as a unit. (3) A block is 512 bytes long. (4) A logical block is 2048 bytes long.

**block file.** A file listing the usage of blocks on a disk.

**block special file.** A special file that provides access to an input or output device is capable of supporting a file system. See also *character special file*.

---

**bootstrap.** A small program that loads larger programs during system initialization.

**branch.** In a computer program an instruction that selects one of two or more alternative sets of instructions. A conditional branch occurs only when a specified condition is met.

**breakpoint.** A place in a computer program, usually specified by an instruction, where execution may be interrupted by external intervention or by a monitor program.

**buffer.** (1) A temporary storage unit, especially one that accepts information at one rate and delivers it at another rate. (2) An area of storage, temporarily reserved for performing input or output, into which data is read, or from which data is written.

**burst pages.** On continuous-form paper, pages of output that can be separated at the perforations.

**byte.** The amount of storage required to represent one character; a byte is 8 bits.

**call.** (1) To activate a program or procedure at its entry point. Compare with *load*.

**callouts.** An AIX kernel parameter establishing the maximum number of scheduled activities that can be pending simultaneously.

**cancel.** To end a task before it is completed.

**carrier return.** (1) In text data, the action causing line ending formatting to be performed at the current cursor location followed by a line advance of the cursor. Equivalent to the carriage return of a typewriter. (2) A keystroke generally indicating the end of a command line.

**case sensitive.** Able to distinguish between uppercase and lowercase letters.

**character.** A letter, digit, or other symbol.

**character display.** A display that uses a character generator to display predefined character boxes of images (characters) on the

screen. This kind of display cannot address the screen any less than one character box at a time. Contrast with *All Points Addressable display*.

**character key.** A keyboard key that allows the user to enter the character shown on the key. Compare with *function keys*.

**character position.** On a display, each location that a character or symbol can occupy.

**character set.** A group of characters used for a specific reason; for example, the set of characters a printer can print or a keyboard can support.

**character special file.** A special file that provides access to an input or output device. The character interface is used for devices that do not use block I/O. See also *block special file*.

**character string.** A sequence of consecutive characters.

**character variable.** The name of a character data item whose value may be assigned or changed while the program is running.

**child.** (1) Pertaining to a secured resource, either a file or library, that uses the user list of a parent resource. A child resource can have only one parent resource. (2) In the AIX Operating System, child is a *process* spawned by a parent process that shares resources of parent process. Contrast with *parent*.

**C language.** A general-purpose programming language that is the primary language of the AIX Operating System.

**class.** Pertaining to the I/O characteristics of a device. AIX devices are classified as block or character.

**client.** In a code service environment, a system that is dependent on a server to provide it with programs or access to programs.

**client partial.** The subset of control information or files in a program install or

---

update distribution that is node unique. That is, those files that must be locally installed on a client for it to successfully run the program in an active-service environment. See also *client*.

**close.** (1) To end an activity and remove that window from the display.

**code.** (1) Instructions for the computer.  
(2) To write instructions for the computer; to *program*. (3) A representation of a condition, such as an error code.

**code page.** In AIX, arrays of code points representing characters that establish ordinal sequence (numeric order) of characters. AIX uses 256-character code pages. Code page P0 consists of 1-byte characters that represent the ASCII, ISO, and EBCDIC character sets and additional characters and symbols. Lower code page P0 (0-127 ordinal) is the ASCII character set. Additional code pages consist of code points for 2-byte character representations.

**code point.** A 1- or 2-byte representation of a character. A byte can contain a single-shifted bit that indicates that the second byte is a part of the same code. In AIX (but not in Japanese Language Support), a byte can contain a single-shifted bit that indicates the code page of the character. Again in AIX, the second byte (only byte in the case of a 1-byte character) places the character in the code page array.

**code segment.** See *segment*.

**code server.** A system that is providing a code service for other computers on a network. See also *code service*.

**code service.** An integrated process where one or more server systems provide access via a Distributed Services network for any number of client systems to the code and functions of AIX and other programs. See also *server* and Distributed Services *network*.

**collating sequence.** The sequence in which characters are ordered within the computer for sorting, combining, or comparing.

**collation.** The process of character and string sorting based on alphabetical order, and, in AIX, on equivalence class. Japanese Language Support uses character class rather than equivalence class.

**color display.** A display device capable of displaying more than two colors and the shades produced via the two colors, as opposed to a monochrome display.

**column.** A vertical arrangement of text or numbers.

**column headings.** Text appearing near the top of columns of data for the purpose of identifying or titling.

**command.** A request to perform an operation or run a program. When parameters, arguments, flags, or other operands are associated with a command, the resulting character string is a single command.

**command interpreter.** A program that sends instructions to the kernel; also called an interface.

**command line.** The area of the screen where commands are displayed as they are typed.

**command line editing keys.** Keys for editing the command line.

**command programming language.** Facility that allows programming by the combination of commands rather than by writing statements in a conventional programming language.

**compile.** (1) To translate a program written in a high-level programming language into a machine language program. (2) The computer actions required to transform a source file into an executable object file.

**compress.** (1) To move files and libraries together on disk to create one continuous area of unused space. (2) In data communications, to delete a series of duplicate characters in a character string.

---

**concatenate.** (1) To link together. (2) To join two character strings.

**condition.** An expression in a program or procedure that can be evaluated to a value of either true or false when the program or procedure is running.

**configuration.** The group of machines, devices, and programs that make up a computer system. See also *system customization*.

**configuration file.** A file that specifies the characteristics of a system or subsystem, for example, the AIX queuing system.

**consistent.** Pertaining to a file system, without internal discrepancies.

**console.** (1) The main AIX display station. (2) A device name associated with the main AIX display station.

**constant.** A data item with a value that does not change. Contrast with *variable*.

**context search.** A search through a file whose target is a character string.

**control block.** A storage area used by a program to hold control information.

**control commands.** Commands that allow conditional or looping logic flow in shell procedures.

**control program.** Part of the AIX Operating System system that determines the order in which basic functions should be performed.

**controlled cancel.** The system action that ends the job step being run, and saves any new data already created. The job that is running can continue with the next job step.

**copy.** The action by which the user makes a whole or partial duplicate of already existing data.

**crash.** An unexpected interruption of computer service, usually due to a serious hardware or software malfunction.

**current directory.** The directory that is active, and can be displayed with the **pwd** command.

**current line.** The line on which the cursor is located.

**current working directory.** See *current directory*.

**cursor.** (1) A movable symbol (such as an underline) on a display, used to indicate to the operator where the next typed character will be placed or where the next action will be directed. (2) A marker that indicates the current data access location within a file.

**cursor movement keys.** The directional keys used to move the cursor.

**customize.** To describe (to the system) the devices, programs, users, and user defaults for a particular data processing system.

**cylinder.** All fixed disk or diskette tracks that can be read or written without moving the disk drive or diskette drive read/write mechanism.

**daemon.** See *daemon process*.

**daemon process.** A process begun by the root or the root shell that can be stopped only by the root. Daemon processes generally provide services that must be available at all times such as sending data to a printer.

**data block.** See *block*.

**data communications.** The transmission of data between computers, or remote devices or both (usually over long distance).

**data stream.** All information (data and control information) transmitted over a data link.

**dbos.** The minimum set of AIX programs that must be present to provide code service.

---

**debug.** (1) To detect, locate, and correct mistakes in a program. (2) To find the cause of problems detected in software.

**default.** A value that is used when no alternative is specified by the operator.

**default directory.** The directory name supplied by the operating system if none is specified.

**default drive.** The drive name supplied by the operating system if none is specified.

**default value.** A value stored in the system that is used when no other value is specified.

**delete.** To remove. For example, to delete a file.

**dependent work station.** A work station having little or no stand alone capability, that must be connected to a host or server in order to provide any meaningful capability to the user.

**device.** An electrical or electronic machine that is designed for a specific purpose and that attaches to your computer, for example, a printer, plotter, disk drive, and so forth.

**device driver.** A program that operates a specific device, such as a printer, disk drive, or display.

**device name.** A name reserved by the system that refers to a specific device.

**diagnostic.** Pertaining to the detection and isolation of an error.

**diagnostic aid.** A tool (procedure, program, reference manual) used to detect and isolate a device or program malfunction or error.

**diagnostic routine.** A computer program that recognizes, locates, and explains either a fault in equipment or a mistake in a computer program.

**digit.** Any of the numerals from 0 through 9.

**directory.** A type of file containing the names and controlling information for other files or other directories.

**disable.** To make nonfunctional.

**discipline.** Pertaining to the order in which requests are serviced, for example, first-come-first-served (fcfs) or shortest job next (sjn).

**disk I/O.** Fixed-disk input and output.

**diskette.** A thin, flexible magnetic plate that is permanently sealed in a protective cover. It can be used to store information copies from the disk or another diskette.

**diskette drive.** The mechanism used to read and write information on diskettes.

**display device.** An output unit that gives a visual representation of data.

**display screen.** The part of the display device that displays information visually.

**display station.** A device that includes a keyboard from which an operator can send information to the system and a display screen on which an operator can see the information sent to or received from the computer.

**Distributed Services.** A licensed program that allows you to use both local and remote directories and files to build file trees.

**Distributed Services network.** A network that is running Distributed Services. See also Distributed Services.

**dump.** (1) To copy the contents of all or part of storage, usually to an output device. (2) Data that has been dumped.

**dump diskette.** A diskette that contains a dump or is prepared to receive a dump.

**dump formatter.** Program for analyzing a dump.

---

**EBCDIC.** See *extended binary-coded decimal interchange code*.

**EBCDIC character.** Any one of the symbols included in the 8-bit EBCDIC set.

**edit.** To modify the form or format of data.

**edit buffer.** A temporary storage area used by an editor.

**editor.** A program used to enter and modify programs, text, and other types of documents and data.

**emulation.** Imitation; for example, when one computer imitates the characteristics of another computer.

**enable.** To make functional.

**enter.** To send information to the computer by pressing the **Enter** key.

**entry.** A single input operation on a work station.

**environment.** The settings for shell variables and paths set associated with each process. These variables can be modified later by the user.

**equivalence class.** In AIX, a grouping of characters (or character strings) that are considered equal for purposes of collation. For example, many languages place an uppercase character in the same equivalence class as its lowercase form, but other languages distinguish between accented and unaccented character forms for the purpose of collation.

**error-correct backspace.** An editing key that performs editing based on a cursor position; the cursor is moved one position toward the beginning of the line, the character at the new cursor location is deleted, and all characters following the cursor are moved one position toward the beginning of the line (to fill the vacancy left by the deleted element).

**escape character.** A character that suppresses the special meaning of one or more characters that follow.

**exit value.** A numeric value that a command returns to indicate whether or not the command completed successfully. Some commands return exit values that give other information, such as whether a file exists. Shell programs can test exit values to control branching and looping.

**expression.** A representation of a value. For example, variables and constants appearing alone or in combination with operators.

**extended binary-coded decimal interchange code (EBCDIC).** A set of 256 eight-bit characters.

**feature.** A programming or hardware option, usually available at an extra cost.

**field.** (1) An area in a record or panel used to contain a particular category of data. (2) The smallest component of a record that can be referred to by a name.

**FIFO.** See *first-in-first-out*.

**file.** A collection of related data that is stored and retrieved by an assigned name.

**file name.** The name used by a program to identify a file. See also *label*.

**filename.** In DOS, that portion of the file name that precedes the extension.

**file specification (filespec).** The name and location of a file. A file specification consists of a drive specifier, a path name, and a file name.

**file system.** The complete structure of directories and files contained on a physical or logical mass storage device, such as a diskette or minidisk.

**filetab.** An AIX kernel parameter establishing the maximum number of files that can be open simultaneously.

---

**file tree.** The complete directory and file structure of a particular node, starting at the root directory. A file tree contains all local and remote mounts performed on minidisks, directories, and files.

**filter.** A command that reads standard input data, modifies the data, and sends it to standard output.

**first-in-first-out (FIFO).** A named permanent pipe. A FIFO allows two unrelated processes to exchange information using a pipe connection.

**first level interrupt handler (FLIH).** A routine that receives control of the system as a result of a hardware interrupt. One FLIH is assigned to each of the six interrupt levels.

**fixed disk.** A flat, circular, nonremoveable plate with a magnetic surface layer on which data can be stored by magnetic recording.

**fixed-disk drive.** The mechanism used to read and write information on fixed disk.

**flag.** A modifier that appears on a command line with the command name that defines the action of the command. Flags in the AIX Operating System almost always are preceded by a dash.

**flattened character.** In AIX, an ASCII character created by translating an extended character to its ASCII equivalent in appearance. Code point information is lost; the character cannot be retranslated to an extended character.

**font.** A family or assortment of characters of a given size and style.

**foreground.** A mode of program execution in which the shell waits for the program specified on the command line to complete before returning your prompt.

**format.** (1) A defined arrangement of such things as characters, fields, and lines, usually

used for displays, printouts, or files. (2) The pattern which determines how data is recorded.

**formatted diskette.** A diskette on which control information for a particular computer system has been written but which may or may not contain any data.

**free list.** A list of available space on each file system. This is sometimes called the free-block list.

**free-block list.** See *free list*.

**full install.** A complete installation of AIX or other programs.

**full path name.** The name of any directory or file expressed as a string of directories and files beginning with the root directory.

**function.** A synonym for procedure. The C language treats a function as a data type that contains executable code and returns a single value to the calling function.

**function keys.** Keys that request actions but do not display or print characters. Included are the keys that normally produce a printed character, but when used with the code key produce a function instead. Compare with *character key*.

**generation.** For some remote systems, the translation of configuration information into machine language.

**Gid.** See *group number*.

**global.** Pertains to information available to more than one program or subroutine.

**global action.** An action having general applicability, independent of the context established by any task.

**global character.** The special characters \* and ? that can be used in a file specification to match one or more characters. For example, placing a ? in a file specification means any character can be in that position.

---

**global search.** The process of having the system look through a document for specific characters, words, or groups of characters.

**global variable.** A symbol defined in one program module, but used in other independently assembled program modules.

**graphic character.** A character that can be displayed or printed.

**group name.** A name that uniquely identifies a group of users to the system.

**group number (Gid).** A unique number assigned to a group of related users. The group number can often be substituted in commands that take a group name as an argument.

**hardware.** The equipment, as opposed to the programming, of a computer system.

**header.** Constant text that is formatted to be in the top margin of one or more pages.

**header label.** A special set of records on a diskette describing the contents of the diskette.

**here document.** Data contained within a shell program or procedure (also called *inline input*).

**highlight.** To emphasize an area on the display by any of several methods, such as brightening the area or reversing the color of characters within the area.

**history file.** A file containing a log of system actions and operator responses.

**hog factor.** In system accounting, an analysis of how many times each command was run, how much processor time and memory it used, and how intensive that use was.

**home directory.** (1) A directory associated with an individual user. (2) The user's current directory on login or after issuing the **cd** command with no argument.

**I/O.** See *input/output*.

**ID.** Identification.

**IF expressions.** Expressions within a procedure, used to test for a condition.

**indirect block.** A block containing pointers to other blocks. Indirect blocks can be single-indirect, double-indirect, or triple-indirect.

**informational message.** A message providing information to the operator, that does not require a response.

**initial program load (IPL).** The process of loading the system programs and preparing the system to run jobs. See *initialize*.

**initialize.** To set counters, switches, addresses, or contents of storage to zero or other starting values at the beginning of, or at prescribed points in, the operation of a computer routine.

**inline input.** See *here document*.

**i-node.** The internal structure for managing files in the system. I-nodes contain all of the information pertaining to the node, type, owner, and location of a file. A table of i-nodes is stored near the beginning of a file system.

**i-number.** A number specifying a particular i-node on a file system.

**inodetab.** A kernel parameter that establishes a table in memory for storing copies of i-nodes for all active files.

**input.** Data to be processed.

**input device.** Physical devices used to provide data to a computer.

**input file.** A file opened by a program so that the program can read from that file.

**input list.** A list of variables to which values are assigned from input data.

**input redirection.** The specification of an input source other than the standard one.

---

**input-output file.** A file opened for input and output use.

**input-output device number.** A value assigned to a device by the virtual machine or to a virtual device by the virtual resource manager. This number uniquely identifies the device regardless of whether it is real or virtual.

**input/output (I/O).** Pertaining to either input, output, or both between a computer and a device.

**input/output subsystem.** That part of the VRM comprised of processes and device managers that provides the mechanism for data transfer and I/O device management and control.

**interactive processing.** A processing method in which each system user action causes response from the program or the system. Contrast with *batch processing*.

**interface.** A shared boundary between two or more entities. An interface might be a hardware component to link two devices together or it might be a portion of storage or registers accessed by two or more computer programs.

**interleave factor.** Specification of the ratio between contiguous physical blocks (on a fixed disk) and logically contiguous blocks (as in a file).

**interrupt.** (1) To temporarily stop a process. (2) In data communications, to take an action at a receiving station that causes the sending station to end a transmission. (3) A signal sent by an I/O device to the processor when an error has occurred or when assistance is needed to complete I/O. An interrupt usually suspends execution of the currently executing program.

**IPL.** See *initial program load*.

**job.** (1) A unit of work to be done by a system. (2) One or more related procedures or programs grouped into a procedure.

**job queue.** A list, on disk, of jobs waiting to be processed by the system.

**justify.** To print a document with even right and left margins.

**kbuffers.** An AIX kernel parameter establishing the number of buffers that can be used by the kernel.

**K-byte.** See *kilobyte*.

**kernel.** A part of the AIX Operating System which participates in the control of computer functions such as input/output, management and control of hardware and software, and scheduling of user processes.

**kernel parameters.** Variables that specify how the kernel allocates certain system resources.

**key pad.** A physical grouping of keys on a keyboard (for example, numeric key pad, and cursor key pad).

**keyboard.** An input device consisting of various keys allowing the user to input data, control cursor and pointer locations, and to control the dialog between the user and the display station

**keylock feature.** A security feature in which a lock and key can be used to restrict the use of the display station.

**keyword.** One of the predefined words of a programming language; a reserved word.

**keyword argument.** One type of variable assignment that can be made on the command line.

**kill.** An AIX Operating System command that stops a process.

---

**kill character.** The character that is used to delete a line of characters entered after the user's prompt.

**kilobyte.** 1024 bytes.

**kprocs.** A kernel parameter establishing the maximum number of processes that the kernel can run simultaneously.

**label.** (1) The name in the disk or diskette volume table of contents that identifies a file. See also *file name*. (2) The field of an instruction that assigns a symbolic name to the location at which the instruction begins, or such a symbolic name.

**left margin.** The area on a page between the left paper edge and the leftmost character position on the page.

**left-adjust.** The process of aligning lines of text at the left margin or at a tab setting such that the leftmost character in the line or filed is in the leftmost position. Contrast with *right-adjust*.

**library.** A collection of functions, calls, subroutines, or other data.

**licensed program product (LPP).** Software programs that remain the property of the manufacturer, for which customers pay a license fee.

**line editor.** An editor that modifies the contents of a file one line at a time.

**linefeed.** An ASCII character that causes an output device to move forward one line.

**link.** A connection between an i-node and one or more file names associated with it.

**literal.** A symbol or a quantity in a source program that is itself data, rather than a reference to data.

**load.** (1) To move data or programs into storage. (2) To place a diskette into a diskette

drive, or a magazine into a diskette magazine drive. (3) To insert paper into a printer.

**loader.** A program that reads run files into main storage, thus preparing them for execution.

**local.** Pertaining to a device directly connected to your system without the use of a communications line. Contrast with *remote*.

**log.** To record; for example, to log all messages on the system printer. A list of this type is called a log, such as an error log.

**log in.** To begin a session at a display station.

**login shell.** The program, or command interpreter, started for a user at login.

**log off.** To end a session at a display station.

**log out.** To end a session at a display station.

**logical device.** A file for conducting input or output with a physical device.

**login user ID.** The ID set by the system for a user at login.

**loop.** A sequence of instructions performed repeatedly until an ending condition is reached.

**main program.** A primary or control program. See also *program*.

**main storage.** The part of the processing unit where programs are run.

**maintenance system.** A special version of the AIX Operating System which is loaded from diskette and used to perform system management tasks.

**major device number.** A system identification number for each device or type of device.

**mapped files.** Files on the fixed disk that are accessed as if they are in memory.

---

**mask.** A pattern of characters that controls the keeping, deleting, or testing of portions of another pattern of characters.

**matrix.** An array arranged in rows and columns.

**maxprocs.** A kernel parameter establishing the maximum number of processes that can be run simultaneously by a user.

**memory.** Storage on electronic chips. Examples of memory are random access memory, read only memory, or registers. See *storage*.

**menu.** A displayed list of items from which an operator can make a selection.

**message.** (1) A response from the system to inform the operator of a condition which may affect further processing of a current program. (2) Information sent from one user in a multiuser operating system to another.

**minidisk.** A logical division of a fixed disk.

**minor device number.** A number used to specify various types of information about a particular device, for example, to distinguish among several printers of the same type.

**mode word.** An i-node field that describes the type and state of the i-node.

**modem.** See *modulator-demodulator*.

**modulation.** Changing the frequency or size of one signal by using the frequency or size of another signal.

**modulator-demodulator (modem).** A device that converts data from the computer to a signal that can be transmitted on a communications line, and converts the signal received to data for the computer.

**module.** (1) A discrete programming unit that usually performs a specific task or set of tasks. Modules are subroutines and calling programs

that are assembled separately, then linked to make a complete program. (2) See *load module*.

**mount.** To make accessible to a file system or file tree. AIX allows local file and directory mounts. Distributed Services permits those file mounts to occur for a remote node.

**mountab.** A kernel parameter establishing the maximum number of file systems that can be mounted simultaneously.

**multiprogramming.** The processing of two or more programs at the same time.

**multivolume file.** A diskette file occupying more than one diskette.

**nest.** To incorporate a structure or structures of some kind into a structure of the same kind. For example, to nest one loop (the nested loop) within another loop (the nesting loop); to nest one subroutine (the nested subroutine) within another subroutine (the nesting subroutine).

**network.** A collection of products connected by communication lines for information exchange between locations.

**new-line character.** A control character that causes the print or display position to move to the first position on the next line.

**node.** An individual system connected to a network.

**null.** Having no value, containing nothing.

**null character (NUL).** The character hex 00, used to represent the absence of a printed or displayed character.

**numeric.** Pertaining to any of the digits 0 through 9.

**object code.** Machine-executable instruction, usually generated by a compiler from source code written in a higher level language. Consists of directly executable machine code. For programs that must be linked, object code consists of relocatable machine code.

---

**octal.** A base eight numbering system.

**open.** To make a file available to a program for processing.

**operating system.** Software that directs and controls the hardware and software in the computer system in which the operating system resides, by providing services such as resource allocation, scheduling, input/output control, and data management.

**operation.** A specific action (such as move, add, multiply, load) that the computer performs when requested.

**operator.** A symbol representing an operation to be done.

**output.** The result of processing data.

**output devices.** Physical devices used by a computer to present data to a user.

**output file.** A file that is opened by a program so that the program can write to that file.

**output redirection.** The specification of an output destination other than the standard one.

**overmount.** A mount in which the path to the mount point and the path to the mounted object are the same. For example in a code server environment you might mount a server's `/usr/lib` over a client's `/usr/lib`.

**override.** (1) A parameter or value that replaces a previous parameter or value. (2) To replace a parameter or value.

**overwrite.** To write output into a storage or file space that is already occupied by data.

**owner.** The user who has the highest level of access authority to a data object or action, as defined by the object or action.

**pad.** To fill unused positions in a field with dummy data, usually zeros or blanks.

**page.** A block of instructions, data, or both.

**page space minidisk.** The area on a fixed disk that temporarily stores instructions or data currently being run. See also *minidisk*.

**pagination.** The process of adjusting text to fit within margins and/or page boundaries.

**paging.** The action of transferring instructions, data, or both between real storage and external page storage.

**parallel processing.** The condition in which multiple tasks are being performed simultaneously within the same activity.

**parameter.** Information that the user supplies to a panel, command, or function.

**parent.** Pertaining to a secured resource, either a file or library, whose user list is shared with one or more other files or libraries. Contrast with *child*.

**parent directory.** The directory one level above the current directory.

**partition.** See *minidisk*.

**passive service.** A code service relationship between a client and a server where the client accesses a server, installs one or more programs from files on the server then disconnects and runs as a stand-alone system. Contrast with *active service*. See also *client* and *server*.

**password.** A string of characters that, when entered along with a user identification, allows an operator to sign on to the system.

**password security.** A program product option that helps prevent the unauthorized use of a display station, by checking the password entered by each operator at sign-on.

**path name.** See *full path name* and *relative path name*.

**pattern-matching character.** Special characters such as `*` or `?` that can be used in search patterns. Some used in a file specification to match one or more characters.

---

For example, placing a ? in a file specification means any character can be in that position. Pattern-matching characters are also called wildcards.

**permission code.** A three-digit octal code, or a nine-letter alphabetic code, indicating the access permissions. The access permissions are read, write, and execute.

**permission field.** One of the three-character fields within the permissions column of a directory listing indicating the read, write, and run permissions for the file or directory owner, group, and all others.

**phase.** One of several stages file system checking and repair performed by the **fsck** command.

**physical device.** See *device*.

**physical file.** An indexed file containing data for which one or more alternative indexes have been created.

**physical record.** (1) A group of records recorded or processed as a unit. Same as *block*. (2) A unit of data moved into or out of the computer.

**PID.** See *process ID*.

**pipe.** To direct the data so that the output from one process becomes the input to another process.

**pipeline.** A direct, one-way connection between two or more processes.

**pitch.** A unit of width of typewriter type, based on the number of times a letter can be set in a linear inch. For example, 10-pitch type has 10 characters per inch.

**platen.** The support mechanism for paper on a printer, commonly cylindrical, against which printing mechanisms strike to produce an impression.

**pointer.** A logical connection between physical blocks.

**port.** (1) To make the programming changes necessary to allow a program that runs on one type of computer to run on another type of computer. (2) An access point for data input to or data output from a computer system. See *connector*.

**position.** The location of a character in a series, as in a record, a displayed message, or a computer printout.

**positional parameter.** A shell facility for assigning values from the command line to variables in a program.

**print queue.** A file containing a list of the names of files waiting to be printed.

**printout.** Information from the computer produced by a printer.

**priority.** The relative ranking of items. For example, a job with high priority in the job queue will be run before one with medium or low priority.

**priority number.** A number that establishes the relative priority of printer requests.

**privileged instructions.** System control instructions that can only run in the processor's privileged state (VRM mode). Privileged instructions generally manipulate virtual machines or the memory manager; they typically are not used by application programmers. See *privileged state*.

**privileged state.** A hardware protection state in which the processor can run privileged instructions.

**privileged user.** The account with superuser authority.

**problem determination.** The process of identifying why the system is not working. Often this process identifies programs,

---

equipment, data communications facilities, or user errors as the source of the problem.

**problem determination procedure.** A prescribed sequence of steps aimed at recovery from, or circumvention of, problem conditions.

**procedure.** See *shell procedure*.

**process.** (1) A sequence of actions required to produce a desired result. (2) An entity receiving a portion of the processor's time for executing a program. (3) An activity within the system begun by entering a command, running a shell program, or being started by another process.

**process accounting.** An analysis of the use each process makes of the processing unit, memory, and I/O resources.

**process ID (PID).** A unique number assigned to a process that is running.

**profile.** (1) A file containing customized settings for a system or user (2) Data describing the significant features of a user, program, or device.

**program.** A file containing a set of instructions conforming to a particular programming language syntax.

**prompt.** A displayed request for information or operator action.

**propagation time.** The time necessary for a signal to travel from one point on a communications line to another.

**qdaemon.** The daemon process that maintains a list of outstanding jobs and sends them to the specified device at the appropriate time.

**queue.** A line or list formed by items waiting to be processed.

**queued message.** A message from the system that is added to a list of messages stored in a file for viewing by the user at a later time. This

is in contrast to a message that is sent directly to the screen for the user to see immediately.

**quit.** A key, command, or action that tells the system to return to a previous state or stop a process.

**quote.** To mask the special meaning of certain characters; to cause them to be taken literally.

**random access.** An access mode in which records can be read from, written to, or removed from a file in any order.

**readonly.** Pertaining to file system mounting, a condition that allows data to be read, but not modified.

**recovery procedure.** (1) An action performed by the operator when an error message appears on the display screen. Usually, this action permits the program to continue or permits the operator to run the next job. (2) The method of returning the system to the point where a major system error occurred and running the recent critical jobs again.

**redirect.** To divert data from a process to a file or device to which it would not normally go.

**reference count.** In an i-node, a record of the total number of directory entries that refer to the i-node.

**relational expression.** A logical statement describing the relationship (such as greater than or equal) of two arithmetic expressions or data items.

**relational operator.** The reserved words or symbols used to express a relational condition or a relational expression.

**relative address.** An address specified relative to the address of a symbol. When a program is relocated, the addresses themselves will change, but the specification of relative addresses remains the same.

---

**relative addressing.** A means of addressing instructions and data areas by designating their locations relative to some symbol.

**relative path name.** The name of a directory or file expressed as a sequence of directories followed by a file name, beginning from the current directory.

**remote.** Pertaining to a system or device that is connected to your system through a communications line. Contrast with *local*.

**reserved character.** A character or symbol that has a special (nonliteral) meaning unless quoted.

**reserved word.** A word that is defined in a programming language for a special purpose, and that must not appear as a user-declared identifier.

**reset.** To return a device or circuit to a clear state.

**restore.** To return to an original value or image. For example, to restore a library from diskette.

**right adjust.** The process of aligning lines of text at the right margin or tab setting such that the rightmost character in the line or file is in the rightmost position.

**right justify.** See right align.

**right margin.** The area on a page between the last text character and the right upper edge.

**right-adjust.** To place or move an entry in a field so that the rightmost character of the field is in the rightmost position. Contrast with *left-adjust*.

**root.** Another name sometimes used for superuser.

**root directory.** The top level of a tree-structured directory system.

**root file system.** The basic AIX Operating System file system, which contains operating

system files and onto which other file systems can be mounted. The root file system is the file system that contains the files that are run to start the system running.

**routine.** A set of statements in a program causing the system to perform an operation or a series of related operations.

**run.** To cause a program, utility, or other machine function to be performed.

**run-time environment.** A collection of subroutines and shell variables that provide commonly used functions and information for system components.

**scratch file.** A file, usually used as a work file, that exists until the program that uses it ends.

**screen.** See *display screen*.

**scroll.** To move information vertically or horizontally to bring into view information that is outside the display screen boundaries.

**second level interrupt handler (SLIH).** A routine that handles the processing of an interrupt from a specific adapter. An SLIH is called by the first level interrupt handler associated with that interrupt level.

**sector.** (1) An area on a disk track or a diskette track reserved to record information. (2) The smallest amount of information that can be written to or read from a disk or diskette during a single read or write operation.

**security.** The protection of data, system operations, and devices from accidental or intentional ruin, damage, or exposure.

**segment.** A contiguous area of virtual storage allocated to a job or system task. A program segment can be run by itself, even if the whole program is not in main storage.

**separator.** A character used to separate parts of a command or file.

---

**sequential access.** An access method in which records are read from, written to, or removed from a file based on the logical order of the records in the file.

**server.** (1) On a network, the computer that contains programs, data, or provides the facilities to be accessed by other computers on the network. (2) A program that handles protocol, queueing, routing, and other tasks necessary for data transfer between devices in a computer system. (3) An application program that usually runs in the background (daemon) and is controlled by the System Program Controller.

**session records.** In the accounting system, a record of time connected and line usage for connected display stations, produced from login and logoff records.

**set flags.** Flags that can be put into effect with the shell set command.

**shared printer.** A printer that is used by more than one work station.

**shell.** A program that accepts and interprets commands for the operating system, such as sh, csh, and the DOS shell program. Also called a *shell program*.

**shell procedure.** A series of commands contained in a file that carry out a particular function when the file is run or when the file is specified as an argument to the sh command. Also called a *shell script*.

**shell program.** See *shell*.

**shell prompt.** The character string on the command line indicating that the system can accept a command (typically the \$ character).

**shell script.** See *shell procedure*.

**shell variables.** Facilities of the shell program for assigning variable values to constant names.

**shutdown.** The process of ending the operation of a system or a subsystem by following a defined procedure.

**size field.** In an i-node, a field that indicates the size, in bytes, of the file associated with the i-node.

**software.** Programs.

**sort.** To rearrange some or all of a group of items based upon the contents or characteristics of those items.

**source diskette.** The diskette containing data to be copied, compared, restored, or backed up.

**source program.** A set of instructions written in a programming language, that must be translated to machine language compiled before the program can be run.

**special character.** A character other than an alphabetic or numeric character. For example; \*, +, and % are special characters.

**special file.** Used in the AIX Operating System to provide an interface to input/output devices. There is at least one special file for each device connected to the computer. Contrast with *directory* and *file*. See also *block special file* and *character special file*.

**spool files.** Files used in the transmission of data among devices.

**standalone shell.** A limited version of the shell program used for system maintenance.

**stand-alone system.** See *stand-alone work station*.

**stand-alone work station.** (1) A work station that can be used to perform tasks independent of (without being connected to) other resources such as servers or host systems. (2) A node that either does not have Distributed Services installed or is acting in ways that do not use the function provided by Distributed Services.

---

**standard error (STDERR).** The place where many programs place error messages.

**standard input (STDIN).** The primary source of data going into a command. Standard input comes from the keyboard unless redirection or piping is used, in which case standard input can be from a file or the output from another command.

**standard output (STDOUT).** The primary destination of data coming from a command. Standard output goes to the display unless redirection or piping is used, in which case standard output can be to a file or another command.

**stanza.** A group of lines in a file that together have a common function. Stanzas are usually separated by blank lines, and each stanza has a name.

**statement.** An instruction in a program or procedure.

**status.** (1) The current condition or state of a program or device. For example, the status of a printer. (2) The condition of the hardware or software, usually represented in a status code.

**STDERR.** See *standard error*.

**STDIN.** See *standard input*.

**STDOUT.** See *standard output*.

**storage.** (1) The location of saved information. (2) In contrast to memory, the saving of information on physical devices such as disk or tape. See *memory*.

**storage device.** A device for storing and/or retrieving data.

**string.** A linear sequence of entities such as characters or physical elements. Examples of strings are alphabetic string, binary element string, bit string, character string, search string, and symbol string.

**su.** See *superuser*.

**subdirectory.** A directory contained within another directory in the file system hierarchy.

**subprogram.** A program invoked by another program. Contrast with *main program*.

**subroutine.** (1) A sequenced set of statements that may be used in one or more computer programs and at one or more points in a computer program. (2) A routine that can be part of another routine. See also *routine*.

**subscript.** An integer or variable whose value refers to a particular element in a table or an array.

**subshell.** An instance of the shell program started from an existing shell program.

**substring.** A part of a character string.

**subsystem.** A secondary or subordinate system, usually capable of operating independently of, or synchronously with, a controlling system.

**superblock.** The most critical part of the file system containing information about every allocation or deallocation of a block in the file system.

**superuser (su).** The user who can operate without the restrictions designed to prevent data loss or damage to the system (user ID 0).

**superuser authority.** The unrestricted ability to access and modify any part of the operating system that is associated with the user who manages the system. The authority obtained when one logs in as **root**.

**system.** The computer and its associated devices and programs.

**system call.** A request by an active process for a service by the system kernel.

**system customization.** A process of specifying the devices, programs, and users for a particular data processing system.

---

**system date.** The date assigned by the system user during setup and maintained by the system.

**system dump.** A copy of memory made whenever an error stops the system. Contrast with *task dump*.

**system management.** The tasks involved in maintaining the system in good working order and modifying the system to meet changing requirements.

**system parameters.** See *kernel parameters*.

**system profile.** A file containing the default values used in system operations.

**system unit.** The part of the system that contains the processing unit, the disk drives, and the diskette drives.

**system user.** A person who uses a computer system.

**target diskette.** The diskette to be used to receive data from a source diskette.

**task.** A basic unit of work to be performed. Examples are a user task, a server task, and a processor task.

**task dump.** A copy of memory associated program that failed (and its data). Contrast with *system dump*.

**terminal.** An input/output device containing a keyboard and either a display device or a printer. Terminals usually are connected to a computer and allow a person to interact with the computer.

**text.** A type of data consisting of a set of linguistic characters (for example, alphabet, numbers, and symbols) and formatting controls.

**text application.** A program defined for the purpose of processing text data (for example, memos, reports, and letters).

**text editing program.** See *editor* and *text application*.

**texttab.** A kernel parameter establishing the size of the text table, in memory, that contains one entry each active shared program text segment.

**trace.** To record data that provides a history of events occurring in the system.

**trace table.** A storage area into which a record of the performance of computer program instructions is stored.

**track.** A circular path on the surface of a fixed disk or diskette on which information is magnetically recorded and from which recorded information is read.

**trap.** An unprogrammed, hardware-initiated jump to a specific address. Occurs as a result of an error or certain other conditions.

**tree-structured directories.** A method for connecting directories such that each directory is listed in another directory except for the root directory, which is at the top of the tree.

**truncate.** To shorten a field or statement to a specified length.

**trusted communications path.** A secure path to the system, invoked with a key sequence and used when entering or changing security-relevant information in the system. Used, for example, when changing passwords or logging in to the system.

**trusted computing base.** The total of all system components, both hardware and software, that protect data in the system.

**trusted program.** A program which assures proper function and is known to be free of programs that can compromise security.

**trusted shell.** A modified command interpreter that provides a restricted environment to perform administrative tasks in a secure manner.

---

**typematic key.** A key that repeats its function multiple times when held down.

**typestyle.** Characters of a given size, style and design.

**Uid.** See *user number*.

**update.** An improvement for some part of the system.

**user.** The name associated with an account.

**user account.** See *account*.

**user ID.** See *user number*.

**user name.** A name that uniquely identifies a user to the system.

**user number (Uid).** (1) A unique number identifying an operator to the system. This string of characters limits the functions and information the operator is allowed to use. The Uid can often be substituted in commands that take a user's name as an argument.

**user profile.** A file containing a description of user characteristics and defaults (for example, printer assignment, formats, group ID) to be conveyed to the system while the user is signed on.

**utility.** A service; in programming, a program that performs a common service function.

**valid.** (1) Allowed. (2) True, in conforming to an appropriate standard or authority.

**value.** (1) In Usability Services, information selected or typed into a pop-up. (2) A set of characters or a quantity associated with a parameter or name. (3) In programming, the contents of a storage location.

**variable.** A name used to represent a data item whose value can change while the program is running. Contrast with *constant*.

**verify.** To confirm the correctness of something.

**version.** Information in addition to an object's name that identifies different modification levels of the same logical object.

**virtual device.** A device that appears to the user as a separate entity but is actually a shared portion of a real device. For example, several virtual terminals may exist simultaneously, but only one is active at any given time.

**virtual machine.** The hardware-independent portion (kernel, shells, libraries, and other subsystems) of the AIX Operating System and user applications.

**virtual machine interface (VMI).** A standard software interface between the kernel and the Virtual Resource Manager.

**virtual resource manager (VRM).** A portion of the AIX Operating System that provides various services, interfaces and run-time routines, through which AIX controls the IBM RT hardware and peripherals.

**virtual resources.** See *virtual resource manager*.

**virtual storage.** Addressable space that appears to be real storage. From virtual storage, instructions and data are mapped into real storage locations.

**virtual terminal.** Any of several logical equivalents of a display station available at a single physical display station.

**Volume ID (Vol ID).** A series of characters recorded on the diskette used to identify the diskette to the user and to the system.

**VRM.** See *virtual resource manager*.

**wildcard.** See *pattern-matching characters*.

**word.** A contiguous series of 32 bits (4 bytes) in storage, addressable as a unit. The address of the first byte of a word is evenly divisible by four.

---

**work file.** A file used for temporary storage of data being processed.

**work station.** A device at which an individual may transmit information to, or receive information from, a computer for the purpose of performing a task, for example, a display

station or printer. See *programmable work station* and *dependent work station*.

**working directory.** See *current directory*.

**wrap around.** Movement of the point of reference in a file from the end of one line to the beginning of the next, or from one end of a file to the other.



---

# Task Index

This index groups commands by task. Each command listing includes a command, a page reference, and a description of the command. Commands are grouped under the following tasks:

Managing the System .....	TASK-3
Installing and Maintaining Programs .....	TASK-3
Configuring the System .....	TASK-3
Backing Up and Restoring System Files .....	TASK-4
Managing File Systems .....	TASK-4
Analyzing System Activity .....	TASK-5
Performing System Accounting Functions .....	TASK-6
Controlling System Security .....	TASK-6
Managing System Auditing .....	TASK-6
Managing the Secure System .....	TASK-6
Managing the Trusted Path .....	TASK-7
Managing Access Permissions and Ownerships .....	TASK-7
Using the System .....	TASK-8
Starting and Stopping the System .....	TASK-8
Using Shells and Interfaces .....	TASK-8
Displaying System Statistics and Information .....	TASK-8
Controlling System Processes .....	TASK-9
Using Disks and Diskettes .....	TASK-10
Using Tape .....	TASK-10
Working with Work Stations .....	TASK-10
Working with Files and Directories .....	TASK-11
Working with Directories .....	TASK-11
Creating and Editing Files .....	TASK-12
Printing and Displaying Files .....	TASK-12
Copying and Moving Files .....	TASK-12
Deleting Files .....	TASK-13
Comparing Files .....	TASK-13
Scanning Files .....	TASK-13
Sorting Files .....	TASK-14
Merging and Splitting Files .....	TASK-14
Working with Remote Files .....	TASK-14
Formatting Text .....	TASK-15
Working with Graphics .....	TASK-15
Protecting Files with File Permissions .....	TASK-16
Backing Up and Restoring Files .....	TASK-16
Working with Data .....	TASK-16
Using Data Tools .....	TASK-16

---

Performing Calculator Functions .....	TASK-17
Communicating on the System .....	TASK-17
Sending Messages and Notices .....	TASK-17
Using Mailboxes .....	TASK-17
Using the MH (Message Handling) Package .....	TASK-17
Communicating with Other Systems .....	TASK-19
Developing Programs .....	TASK-19
Programming in Assembler .....	TASK-19
Programming in C .....	TASK-19
Programming in Miscellaneous Languages .....	TASK-20
Programming in Shell .....	TASK-20
Working with Messages .....	TASK-21
Debugging Programs .....	TASK-21
Managing Source Programs Using the Source Code Control System (SCCS) .....	TASK-21
Managing Object Files .....	TASK-22
Playing Games .....	TASK-22

---

## Managing the System

The following groups of commands are important for managing various aspects of the system.

### Installing and Maintaining Programs

<b>cvid</b>	272	Creates a VRM install diskette for backup purposes.
<b>install</b>	524	Installs a command.
<b>installp</b>	529	Installs a licensed program.
<b>make</b>	625	Maintains up-to-date versions of programs.
<b>mdrc</b>	640	Allows you to reinstall a user-created minidisk after you have reinstalled AIX.
<b>ndtable</b>	685	Accesses the Distributed Services Node Table.
<b>pwtable</b>	801	Accesses the Distributed Services Node Security Table.
<b>updatep</b>	1122	Updates one or more programs.

### Configuring the System

<b>bffcreate</b>	108	Creates files in backup format for complete or subset programs in a code service environment.
<b>biodd_cfg</b>	115	Configures the block I/O AIX device driver.
<b>chkcomp</b>	158	Checks compatibility between a code server and an active-service client.
<b>chngstate</b>	164	Changes the state of a code service client to either active-service or stand-alone.
<b>chparm</b>	171	Changes or examines system parameters.
<b>config</b>	194	Extracts configuration information from configuration files.
<b>devices</b>	315	Adds, deletes, changes, and displays device information.
<b>defkey</b>	306	Defines keyboard key assignments.
<b>display</b>	332	Selects the physical display that an existing or new virtual terminal uses and sets colors and fonts.
<b>dsipc</b>	354	Installs the Interprocess Communication key mapping in the kernel.
<b>dsldxprof</b>	355	Loads translate information into the UID/GID translate profiles.
<b>dsxlate</b>	363	Installs Distributed Services UID/GID translate tables into the kernel.
<b>env</b>	393	Sets the environment for execution of a command.
<b>getty</b>	490	Sets the characteristics of ports.
<b>init</b>	521	Initializes the system.
<b>ipctable</b>	544	Creates, displays, or changes the Distributed Services IPC Queues Table.
<b>keyboard</b>	551	Controls the delay and repetition rates of the keyboard.
<b>locator</b>	583	Controls the sample rate of the locator.
<b>minidisks</b>	650	Adds, deletes, changes, and displays minidisks.

---

<b>mknod</b>	661	Creates a special file.
<b>pdisable,</b> <b>phold</b>	741	Kills the logger running on the specified port.
<b>pstart,</b> <b>penable,</b> <b>pshare,</b> <b>pdelay</b>	791	Enables or reports the availability of login ports.
<b>rc</b>	806	Performs normal startup initialization.
<b>setdma</b>	910	Sets the DMA channel of the specified adapter.
<b>sound</b>	967	Controls the volume and click of the keyboard speaker.
<b>splp</b>	975	Changes or displays printer driver settings.
<b>stty</b>	1018	Sets, resets, or reports work station operating parameters.
<b>termdef</b>	1062	Queries terminal characteristics.
<b>trdiag</b>	1097	Starts diagnostics on the Token-Ring Network.
<b>ugtable</b>	1109	Creates, displays, and changes the Distributed Services Network Users/Groups Table.
<b>users,</b> <b>adduser</b>	1129	Adds, deletes, and changes user and group information.
<b>varyon</b>	1180	Makes an external disk drive and any minidisks or file systems defined on it available for use.
<b>varyoff</b>	1177	Removes an external disk drive from the operating system configuration.
<b>verify</b>	1186	Turns write verification on or off for a particular minidisk.
<b>vrmconfig</b>	1206	Installs peripheral devices.

## Backing Up and Restoring System Files

<b>backup</b>	88	Backs up files.
<b>pack</b>	730	Compresses files.
<b>restore</b>	826	Copies back files created by the <b>backup</b> command.
<b>tapechk</b>	1047	Performs consistency checking of the streaming tape device.
<b>tar</b>	1048	Manipulates archives.
<b>tctl</b>	1058	Gives commands to streaming tape.

## Managing File Systems

<b>basename</b>	95	Returns the base name of a string parameter.
<b>chroot</b>	172	Changes the root directory of a command.
<b>clri</b>	175	Clears the specified i-node.
<b>cpio</b>	205	Copies files into and out of archive storage and directories.
<b>crash</b>	215	Examines system images.

---

<b>dcopy</b>	299	Copies file systems for the best access time.
<b>devnm</b>	316	Names a device.
<b>df</b>	318	Reports number of available disk blocks.
<b>env</b>	393	Sets the environment for execution of a command.
<b>ff</b>	417	Lists the file names and statistics for a file system.
<b>fsck</b>	445	Checks file system consistency and interactively repairs the file system.
<b>fsdb</b>	450	Debugs file systems.
<b>istat</b>	545	Examines i-nodes.
<b>link</b>	575	Performs a link or unlink system call.
<b>mkfs</b>	658	Makes a file system.
<b>mknod</b>	661	Creates a special file.
<b>mount</b>	669	Makes a file system available for use.
<b>ncheck</b>	683	Generates path names from i-numbers.
<b>proto</b>	780	Constructs a prototype file for a file system.
<b>setmnt</b>	911	Creates mount table.
<b>skulker</b>	951	Cleans up file systems by removing unwanted files.
<b>sync</b>	1030	Updates the superblock and writes buffered files to the fixed disk.
<b>umount,</b> <b>unmount</b>	1112	Unmounts a previously mounted file system, directory, or file.
<b>varyon</b>	1180	Makes an external disk drive and any minidisks or file systems defined on it available for use.
<b>varyoff</b>	1177	Removes an external disk drive from the operating system configuration.

## Analyzing System Activity

<b>du</b>	364	Summarizes disk usage.
<b>dumpfmt</b>	368	Formats the VRM dump file.
<b>errdead</b>	397	Extracts error records from dump.
<b>errdemon</b>	398	Starts the error-logging daemon.
<b>errpt</b>	400	Processes a report of logged errors.
<b>errstop</b>	404	Terminates the error-logging daemon.
<b>errupdate</b>	405	Updates an error report template.
<b>fuser</b>	455	Identifies processes using a file or file structure.
<b>profiler</b>	775	Profiles the operating system.
<b>time</b>	1068	Times the execution of a command.
<b>trace</b>	1086	Starts the trace function.
<b>trcrpt</b>	1091	Formats a report from the trace log file.
<b>trcstop</b>	1093	Stops the trace function.
<b>trcupdate</b>	1094	Updates trace format templates.

---

## Performing System Accounting Functions

<b>acct/*</b>	13	Provides accounting shell procedures.
<b>acctcms</b>	18	Produces command usage summaries from accounting records.
<b>acctcom</b>	20	Displays selected process accounting record summaries.
<b>acctcon</b>	24	Performs connect-time accounting.
<b>acctdisk,</b>	26	Performs disk-usage accounting.
<b>acctdusg</b>		
<b>acctmerg</b>	28	Merges total accounting files.
<b>acctprc</b>	30	Performs process accounting.
<b>diskusg</b>	330	Generates disk accounting data by user ID.
<b>fwtmp</b>	457	Manipulates connect accounting records.
<b>runacct</b>	848	Runs daily accounting.
<b>sadc</b>	863	Provides a system activity report package.

## Controlling System Security

The following groups of commands are important to ensure the security of the system.

### Managing System Auditing

<b>audit</b>	67	Controls system auditing.
<b>auditapp</b>	69	Adds an audit bin file to the end of the audit trail file.
<b>auditbin</b>	71	Manages bins of audit information.
<b>auditpr</b>	73	Displays audit trail files.
<b>auditselect</b>	76	Selects audit records.
<b>auditstream</b>	78	Creates a channel for the reading of audit records.
<b>auditwrite</b>	80	Generates an audit record at the command level.

### Managing the Secure System

<b>init</b>	521	Initializes the system.
<b>login</b>	584	Allows you to sign on to the system and performs user identification and authentication.
<b>loginx</b>	587	Sets up a user's execution environment.
<b>passwd</b>	735	Changes login password.
<b>print</b>	767	Enqueues a file.
<b>secure</b>	885	Establishes a more secure system configuration
<b>shell</b>	938	Executes a shell in a user's login environment.

---

<b>sysck</b>	1031	Verifies the secure system state.
<b>tsh</b>	1100	Interprets commands in a trusted shell.
<b>tvi</b>	1108	Acts as trusted editor for system administration.
<b>watch</b>	1209	Observes and reports security-relevant actions on the system.

## Managing the Trusted Path

<b>actman</b>	32	Permits interaction with multiple virtual terminals.
<b>getty</b>	490	Sets the characteristics of ports.
<b>init</b>	521	Initializes the system.
<b>login</b>	584	Allows you to sign on to the system and performs user identification and authentication.
<b>loginx</b>	587	Sets up a user's execution environment.
<b>logout</b>	590	Stops all processes on a port, returning it to a dead state.
<b>shell</b>	938	Executes a shell in a user's login environment.
<b>tsh</b>	1100	Interprets commands in a trusted shell.
<b>tvi</b>	1108	Acts as trusted editor for system administration

## Managing Access Permissions and Ownerships

<b>chgrp</b>	156	Changes the group ownership of a file or directory.
<b>chmod</b>	160	Changes permission codes.
<b>chown</b>	169	Changes the owner of files or directories.
<b>chtcb</b>	174	Sets or queries the tcb attribute of a file.
<b>groups</b>	506	Displays your group membership.
<b>id</b>	517	Displays the system identity of the user issuing the command.
<b>logname</b>	589	Displays your login name.
<b>login</b>	584	Allows you to sign on to the system and performs user identification and authentication.
<b>makekey</b>	634	Generates an encryption key.
<b>newgrp</b>	689	Changes your primary group identification.
<b>passwd</b>	735	Changes login password.
<b>pwck</b>	798	Checks the password and group files for inconsistencies.
<b>su</b>	1026	Obtains the privileges of another user, including superuser authority.
<b>umask</b>	1110	Displays and sets file-creation permission code mask.
<b>users, adduser</b>	1129	Adds, deletes, and changes user and group information.

---

## Using the System

The following groups of commands help you use the various functions of the system.

### Starting and Stopping the System

<b>actman</b>	32	Permits interaction with multiple virtual terminals.
<b>login</b>	584	Allows you to sign on to the system and performs user identification and authentication.
<b>open</b>	728	Opens a virtual terminal.
<b>passwd</b>	735	Changes login password.
<b>shutdown</b>	946	Ends system operation.

### Using Shells and Interfaces

<b>actman</b>	32	Permits interaction with multiple virtual terminals.
<b>csh</b>	225	Interprets commands read from a file or entered from the keyboard.
<b>dos</b>	341	Starts <b>shell</b> .
<b>sh</b>	913	Interprets commands read from a file or entered at the keyboard.
<b>tsh</b>	1100	Interprets commands in a trusted shell.

### Displaying System Statistics and Information

<b>date</b>	281	Displays or sets the date.
<b>devices</b>	315	Adds, deletes, changes, and displays device information.
<b>diskusg</b>	330	Generates disk accounting data by user ID.
<b>dsstate</b>	361	Sets the state of the Distributed Services kernel logic.
<b>errpt</b>	400	Processes a report of logged errors.
<b>errupdate</b>	405	Updates an error report template.
<b>file</b>	420	Determines file type.
<b>fptype</b>	444	Displays the floating point configuration of the system.
<b>fuser</b>	455	Identifies processes using a file or file structure.
<b>groups</b>	506	Displays your group membership.
<b>help</b>	513	Provides information about a Source Code Control System (SCCS) message or command or about certain non-SCCS commands.
<b>id</b>	517	Displays the system identity of the user issuing the command.
<b>ipcs</b>	539	Reports interprocess communication facility status.
<b>ipctable</b>	544	Creates, displays, or changes the Distributed Services IPC Queues Table.
<b>istat</b>	545	Examines i-nodes.

---

<b>logname</b>	589	Displays your login name.
<b>man</b>	635	Displays manual entries online.
<b>minidisks</b>	650	Adds, deletes, changes, and displays minidisks.
<b>ncheck</b>	683	Generates path names from i-numbers.
<b>ndtable</b>	685	Accesses the Distributed Services Node Table.
<b>news</b>	691	Writes system news items to standard output.
<b>od</b>	723	Writes the contents of storage to the standard output.
<b>profiler</b>	775	Profiles the operating system.
<b>PostScript</b>	786	Reports process status.
<b>pstart,</b> <b>penable,</b> <b>pshare,</b> <b>pdelay</b>	791	Enables or reports the availability of login ports.
<b>pwck</b>	798	Checks the password and group files for inconsistencies.
<b>pwd</b>	800	Displays the path name of the working directory.
<b>pwtable</b>	801	Accesses the Distributed Services Node Security Table.
<b>rstatd</b>	847	Returns NFS performance statistics from the kernel.
<b>rup</b>	854	Displays the host status of local machines.
<b>rusers</b>	856	Identifies users logged in on network hosts.
<b>sact</b>	862	Displays current Source Code Control System (SCCS) file editing status.
<b>sadc</b>	863	Provides a system activity report package.
<b>sag</b>	865	Displays a graph of system activity.
<b>sar</b>	867	Collects, reports, or saves system activity information.
<b>splp</b>	975	Changes or displays printer driver settings.
<b>stty</b>	1018	Sets, resets, or reports work station operating parameters.
<b>sum</b>	1029	Displays the checksum and block count of a file.
<b>time</b>	1068	Times the execution of a command.
<b>timex</b>	1069	Times a command, and reports process data and system activity.
<b>tty</b>	1105	Writes to standard output the full path name of your work station.
<b>ugtable</b>	1109	Creates, displays, and changes the Distributed Services Network Users/Groups Table.
<b>uname</b>	1114	Displays the name of the current operating system.
<b>uustat</b>	1158	Reports the status of and provides rudimentary job control for BNU commands.
<b>who</b>	1219	Identifies the users currently logged in.

## Controlling System Processes

<b>at</b>	63	Runs commands at a later time.
<b>cron</b>	220	Runs commands automatically.
<b>crontab</b>	222	Submits a schedule of commands to <b>cron</b> .

---

<b>errdemon</b>	398	Starts the error-logging daemon.
<b>errstop</b>	404	Terminates the error-logging daemon.
<b>kill</b>	552	Sends a signal to a running process.
<b>killall</b>	555	Cancels all processes except the calling process.
<b>nice</b>	699	Runs a command at a different priority.
<b>nohup</b>	707	Runs a command without hangups and quits.
<b>open</b>	728	Opens a virtual terminal.
<b>qdaemon</b>	802	Schedules jobs enqueued by the <b>print</b> command.
<b>sleep</b>	952	Suspends execution for an interval.
<b>syslogd</b>	1037	Reads and logs messages.
<b>tlog</b>	1071	Stops or restarts sending of terminal I/O to a daemon.
<b>tlogger</b>	1072	Gathers I/O from a terminal and writes it to a log file.
<b>writesrv</b>	1230	Allows Distributed Services users to send messages to and receive messages from a remote system.

## Using Disks and Diskettes

<b>format</b>	436	Formats diskettes.
<b>mdrc</b>	640	Allows you to reinstall a user-created minidisk after you have reinstalled AIX.
<b>minidisks</b>	650	Adds, deletes, changes, and displays minidisks.
<b>mount</b>	669	Makes a file system available for use.
<b>umount,</b> <b>unmount</b>	1112	Unmounts a previously mounted file system, directory, or file.
<b>varyon</b>	1180	Makes an external disk drive and any minidisks or file systems defined on it available for use.
<b>varyoff</b>	1177	Removes an external disk drive from the operating system configuration.
<b>verify</b>	1186	Turns write verification on or off for a particular minidisk.

## Using Tape

<b>tapechk</b>	1047	Performs consistency checking of the streaming tape device.
<b>tar</b>	1048	Manipulates archives.
<b>tctl</b>	1058	Gives commands to streaming tape.

## Working with Work Stations

<b>defkey</b>	306	Defines keyboard key assignments.
<b>display</b>	332	Selects the physical display that an existing or new virtual terminal uses and sets colors and fonts.

---

<b>echo</b>	369	Writes its arguments to standard output.
<b>hp</b>	514	Handles special functions for the HP2640- and HP2621-series terminals.
<b>keyboard</b>	551	Controls the delay and repetition rates of the keyboard.
<b>locator</b>	583	Controls the sample rate of the locator.
<b>pdisable,</b>	741	Kills the logger running on the specified port.
<b>phold</b>		
<b>pstart,</b>	791	Enables or reports the availability of login ports.
<b>penable,</b>		
<b>pshare,</b>		
<b>pdelay</b>		
<b>stty</b>	1018	Sets, resets, or reports work station operating parameters.
<b>tabs</b>	1041	Sets tab stops on work stations.
<b>termdef</b>	1062	Queries terminal characteristics.
<b>tic</b>	1067	Translates <b>terminfo</b> files from source to compiled format.
<b>tput</b>	1081	Queries the <b>terminfo</b> file.
<b>tty</b>	1105	Writes to standard output the full path name of your work station.
<b>300</b>	1262	Handles special line-motion functions for DASI 300/300s work stations.
<b>4014</b>	1264	Formats a full page 66-line screen display for a Tektronix 4014 work station.
<b>450</b>	1265	Handles special line-motion functions for the DASI 450 work station.

## Working with Files and Directories

The following groups of commands allow you to create and manipulate files and directories.

### Working with Directories

<b>cd</b>	150	Changes the current directory.
<b>chroot</b>	172	Changes the root directory of a command.
<b>dircmp</b>	328	Compares two directories and the contents of their common files.
<b>dosdir</b>	346	Lists the directory for DOS files.
<b>find</b>	422	Finds files matching expression.
<b>li</b>	567	Lists the contents of a directory.
<b>ls</b>	595	Displays the contents of a directory.
<b>mkdir</b>	657	Makes a directory.
<b>mvdir</b>	682	Moves (renames) a directory.
<b>pwd</b>	800	Displays the path name of the working directory.
<b>rm</b>	833	Removes files or directories.
<b>rmdir</b>	838	Removes a directory.

---

## Creating and Editing Files

<b>admin</b>	41	Creates and initializes SCCS files.
<b>cdc</b>	152	Changes the comments in a Source Code Control System (SCCS) delta.
<b>ed</b>	371	Edits text by line.
<b>edit</b>	387	Provides a simple line editor for the new user.
<b>ex</b>	407	Edits lines interactively, with screen display.
<b>get</b>	477	Creates a specified version of a Source Code Control System (SCCS) file.
<b>mknod</b>	661	Creates a special file.
<b>sed</b>	887	Provides a stream editor.
<b>spell</b>	969	Finds spelling errors.
<b>tab</b>	1040	Changes spaces into tabs or tabs into spaces.
<b>uniq</b>	1118	Deletes repeated lines in a file.
<b>vi</b>	1187	Edits files with a full screen display.

## Printing and Displaying Files

<b>cat</b>	137	Concatenates or displays files.
<b>cut</b>	269	Writes out selected fields from each line of a file.
<b>lp</b>	593	Prints a file in a format suitable for sending to a line printer.
<b>nl</b>	701	Numbers lines in a file.
<b>od</b>	723	Writes the contents of storage to the standard output.
<b>pg</b>	744	Formats files to the work station.
<b>piobe</b>	753	Writes a file to standard output in a format suitable for sending to a line printer.
<b>pr</b>	761	Writes a file to standard output.
<b>prs</b>	781	Displays a Source Code Control System (SCCS) file.
<b>print</b>	767	Enqueues a file.
<b>qdaemon</b>	802	Schedules jobs enqueued by the <b>print</b> command.
<b>splp</b>	975	Changes or displays printer driver settings.
<b>tail</b>	1044	Writes a file to standard output, beginning at a specified point.
<b>vc</b>	1182	Substitutes assigned values in place of keywords.

## Copying and Moving Files

<b>cat</b>	137	Concatenates or displays files.
<b>cp</b>	202	Copies files.
<b>Cvt</b>	274	Moves old UUCP files into new BNU directories.
<b>dd</b>	301	Converts and copies a file.
<b>dosread</b>	348	Copies a DOS file.

---

<b>doswrite</b>	350	Copies AIX files to DOS files.
<b>ln</b>	581	Links files.
<b>mv</b>	679	Moves files.
<b>uucp</b>	1144	Copies files from one AIX system to another AIX system.
<b>uuto</b>	1162	Copies public files from one AIX system to another AIX system, with local system control of file access.

## Deleting Files

<b>del</b>	308	Deletes files if the request is confirmed.
<b>dosdel</b>	345	Deletes DOS files.
<b>rm</b>	833	Removes files or directories.
<b>uniq</b>	1118	Deletes repeated lines in a file.
<b>uucleanup</b>	1141	Deletes selected files older than a specified number of hours from the BNU spool directory or a named directory.

## Comparing Files

<b>bdiff</b>	102	Uses <b>diff</b> to find differences in very large files.
<b>cmp</b>	177	Compares two files.
<b>comm</b>	183	Selects or rejects lines common to two sorted files.
<b>diff</b>	320	Compares text files.
<b>diff3</b>	323	Compares three files.
<b>diffmk</b>	326	Marks differences between files.
<b>dircmp</b>	328	Compares two directories and the contents of their common files.
<b>sdiff</b>	883	Compares two files and displays the differences in a side by side format.
<b>scsdiff</b>	874	Compares two versions of a Source Code Control System (SCCS) file.

## Scanning Files

<b>awk</b>	81	Finds lines in files matching specified patterns and performs specified actions on them.
<b>bfs</b>	110	Scans files.
<b>file</b>	420	Determines file type.
<b>find</b>	422	Finds files matching expression.
<b>grep</b>	501	Searches a file for a pattern.
<b>hyphen</b>	516	Finds hyphenated words.
<b>wc</b>	1211	Counts the number of lines, words, and characters in a file.
<b>what</b>	1213	Displays identifying information in files.

---

## Sorting Files

<b>lorder</b>	591	Finds the best order for member files in an object library.
<b>sort</b>	958	Sorts or merges files.
<b>tsort</b>	1102	Sorts an unordered list of ordered pairs (a topological sort).

## Merging and Splitting Files

<b>csplit</b>	252	Splits files by context.
<b>join</b>	547	Joins data fields of two files.
<b>paste</b>	736	Merges the lines of several files or subsequent lines in one file.
<b>sort</b>	958	Sorts or merges files.
<b>split</b>	974	Splits a file into pieces.

## Working with Remote Files

<b>biod</b>	114	Starts daemons that handle NFS block I/O requests.
<b>domainname</b>	340	Displays or sets YP domain name.
<b>makedbm</b>	632	Makes a Yellow Pages <b>dbm</b> map.
<b>mountd</b>	674	Answers NFS mount requests.
<b>nfsd</b>	696	Starts NFS client request daemons.
<b>nfsstat</b>	697	Displays NFS statistics.
<b>on</b>	726	Executes a command remotely via NFS.
<b>pcnfs</b>	739	Serves PC-NFS client requests.
<b>portmap</b>	757	Maps RPC programs to the servicing ports on RPC servers.
<b>rexed</b>	832	Executes remote programs.
<b>rpcgen</b>	843	Compiles a Remote Procedure Call program.
<b>rpcinfo</b>	845	Reports Remote Procedure Call status information.
<b>rstatd</b>	847	Returns NFS performance statistics from the kernel.
<b>rup</b>	854	Displays the host status of local machines.
<b>rusers</b>	856	Identifies users logged in on network hosts.
<b>rusersd</b>	858	Displays list of active NFS users.
<b>rwall</b>	859	Writes to all network users.
<b>rwalld</b>	861	NFS daemon for <b>rwall</b> and <b>shutdown</b> .
<b>showmount</b>	945	Displays list of remotely mounted file systems.
<b>spray</b>	981	Sprays packets to host when NFS is installed.
<b>sprayd</b>	983	Receives packets sent by the <b>spray</b> command.
<b>ypbind</b>	1239	Allows Yellow Pages client processes to communicate with the YP server.
<b>ypcat</b>	1241	Displays values in a Yellow Pages data base.
<b>ypinit</b>	1243	Builds and installs the Yellow Pages data base.

---

<b>ypmatch</b>	1245	Displays the values of one or more keys from a Yellow Pages map.
<b>yppasswd</b>	1247	Changes Yellow Pages passwords
<b>yppasswdd</b>	1249	Handles Yellow Pages password change requests
<b>yppoll</b>	1251	Displays the version of a Yellow Pages map located at a Yellow Pages host.
<b>yppush</b>	1252	<b>yppush</b> .
<b>ypserv</b>	1256	Looks up information in a local Yellow Pages data base.
<b>ypset</b>	1254	Directs the <b>ypbind</b> daemon to a particular server.
<b>ypwhich</b>	1258	Displays the name of the host machine acting as the Yellow Pages server or as a map YP master server.
<b>ypxfr</b>	1260	Transfers a Yellow Pages map to the local host machine.

## Formatting Text

<b>col</b>	179	Processes text having reverse linefeeds and forward/reverse half-linefeeds for output to standard output.
<b>cw</b>	275	Prepares constant-width text for <b>troff</b> .
<b>deroff</b>	313	Removes <b>nroff</b> , <b>troff</b> , <b>tbl</b> , and <b>eqn</b> constructs from files.
<b>eqn</b>	395	Formats mathematical text for the <b>nroff</b> , <b>troff</b> and <b>troff</b> commands.
<b>fmt</b>	428	Formats mail messages prior to sending.
<b>greek</b>	499	Converts output for a Teletype Model 37 work station to output for other work stations.
<b>hp</b>	514	Handles special functions for the HP2640- and HP2621-series terminals.
<b>mm</b>	663	Displays or checks documents formatted with Memorandum Macros.
<b>mmt</b>	666	Typesets documents, manual pages, view graphs, and slides.
<b>newform</b>	686	Changes the format of a text file.
<b>nl</b>	701	Numbers lines in a file.
<b>nroff, troff</b>	709	Formats text for printing devices.
<b>ptx</b>	794	Generates a permuted index.
<b>tbl</b>	1053	Formats tables for the <b>nroff</b> , <b>troff</b> and <b>troff</b> commands.
<b>tc</b>	1056	Simulates phototypesetter output for a Tektronix 4014 work station.
<b>vrm2rtfont</b>	1205	Converts a standard AIX font file to RT rtx font format.
<b>300</b>	1262	Handles special line-motion functions for DASI 300/300s work stations.
<b>4014</b>	1264	Formats a full page 66-line screen display for a Tektronix 4014 work station.
<b>450</b>	1265	Handles special line-motion functions for the DASI 450 work station.

## Working with Graphics

<b>gdev</b>	460	Provides graphical device routines and filters.
<b>ged</b>	463	Displays, makes, and edits graphical files on Tektronix 4010 terminals.
<b>gend</b>	475	Provides a general graphics device backend.

---

<b>graph</b>	494	Draws a graph.
<b>graphics</b>	497	Accesses graphical and numerical commands.
<b>gutil</b>	508	Provides graphical utility programs.
<b>spline</b>	972	Interpolates smooth curve.
<b>stat</b>	984	Provides tools for analyzing numerical data.
<b>toc</b>	1074	Provides graphical table of contents routines.
<b>tplot</b>	1079	Produces plotting instructions for a particular work station.

## Protecting Files with File Permissions

<b>chgrp</b>	156	Changes the group ownership of a file or directory.
<b>chmod</b>	160	Changes permission codes.
<b>chown</b>	169	Changes the owner of files or directories.
<b>groups</b>	506	Displays your group membership.
<b>li</b>	567	Lists the contents of a directory.
<b>ls</b>	595	Displays the contents of a directory.
<b>umask</b>	1110	Displays and sets file-creation permission code mask.

## Backing Up and Restoring Files

<b>ar</b>	55	Maintains portable libraries used by the linkage editor.
<b>backup</b>	88	Backs up files.
<b>cpio</b>	205	Copies files into and out of archive storage and directories.
<b>lorder</b>	591	Finds the best order for member files in an object library.
<b>pack</b>	730	Compresses files.
<b>restore</b>	826	Copies back files created by the <b>backup</b> command.
<b>shlib</b>	939	Creates a shared library.
<b>tar</b>	1048	Manipulates archives.

## Working with Data

The following groups of commands allow you to use various data tools.

### Using Data Tools

<b>banner</b>	94	Writes character strings in large letters to standard output.
<b>cal</b>	132	Displays a calendar.
<b>calendar</b>	134	Writes reminder messages to standard output.
<b>ctab</b>	257	Produces a collating table.

---

<b>echo</b>	369	Writes its arguments to standard output.
<b>tr</b>	1083	Translates characters.
<b>units</b>	1119	Converts units in one measure to equivalent units in another measure.

## Performing Calculator Functions

<b>bc</b>	97	Provides an interpreter for arbitrary-precision arithmetic language.
<b>dc</b>	295	Provides an interactive desk calculator for doing arbitrary-precision integer arithmetic.
<b>factor</b>	416	Factors a number.

## Communicating on the System

The following groups of commands allow you to use mail and message facilities on the system.

### Sending Messages and Notices

<b>confer</b>	189	Provides an online conferencing system.
<b>mesg</b>	642	Permits or refuses <b>write</b> messages.
<b>news</b>	691	Writes system news items to standard output.
<b>wall</b>	1208	Writes a message to all logged-in users.
<b>write</b>	1225	Sends messages to other users on the system.

### Using Mailboxes

<b>bellmail</b>	104	Sends messages to system users and displays messages from system users.
<b>fmt</b>	428	Formats mail messages prior to sending.
<b>mail, Mail</b>	608	Sends and receives mail.
<b>mailstats</b>	623	Displays statistics regarding mail traffic.
<b>sendmail</b>	897	Routes mail for local or network delivery.

### Using the MH (Message Handling) Package

<b>ali</b>	48	Lists mail aliases and their addresses.
<b>anno</b>	50	Annotates messages.
<b>ap</b>	53	Parses and reformats addresses.
<b>burst</b>	129	Explodes digests into messages.
<b>comp</b>	185	Composes a message.

---

<b>conflict</b>	196	Searches for alias and password conflicts.
<b>dist</b>	336	Redistributes a message to additional addresses.
<b>dp</b>	352	Parses and reformats dates.
<b>folder</b>	429	Selects and lists folders and messages.
<b>folders</b>	433	Lists folders and messages.
<b>forw</b>	438	Forwards messages.
<b>inc</b>	518	Incorporates new mail.
<b>install-mh</b>	527	Initializes the MH environment.
<b>mark</b>	637	Creates, modifies, and displays message sequences.
<b>mhl</b>	643	Produces formatted listings of messages.
<b>mhmail</b>	646	Sends or receives mail.
<b>mhpath</b>	648	Prints full path names of messages and folders.
<b>msgchk</b>	675	Checks for messages.
<b>msh</b>	677	Creates an MH shell.
<b>next</b>	694	Shows the next message.
<b>packf</b>	733	Compresses the contents of a folder into a file.
<b>pick</b>	748	Selects messages by content, and creates and modifies sequences.
<b>post</b>	758	Delivers a message.
<b>prev</b>	765	Shows the previous message.
<b>prompter</b>	778	Invokes a prompting editor.
<b>rcvdist</b>	808	Sends a copy of incoming messages to additional recipients.
<b>rcvpack</b>	810	Saves incoming messages in a packed file.
<b>rcvstore</b>	812	Incorporates new mail from standard input into a folder.
<b>rcvtty</b>	815	Notifies the user of incoming messages.
<b>refile</b>	817	Files messages in other folders.
<b>repl</b>	821	Replies to a message.
<b>rmf</b>	839	Removes a folder.
<b>rmm</b>	841	Removes messages.
<b>scan</b>	871	Produces a one line per message scan listing.
<b>send</b>	893	Sends a message.
<b>show</b>	942	Shows messages.
<b>slocal</b>	954	Processes incoming mail.
<b>sortm</b>	965	Sorts messages.
<b>spost</b>	978	Delivers a message.
<b>vmh</b>	1203	Invokes a visual interface for use with MH commands.
<b>whatnow</b>	1215	Invokes a prompting interface for draft disposition.
<b>whom</b>	1222	Lists the addresses of the proposed recipients of a message and verifies the addresses.

---

## Communicating with Other Systems

<b>connect</b>	198	Establishes a connection to a remote system.
<b>ct</b>	254	Dials an attached terminal and issues a login process.
<b>cu</b>	263	Connects directly or indirectly to another UNIX system.
<b>Cvt</b>	274	Moves old UUCP files into new BNU directories.
<b>sum</b>	1029	Displays the checksum and block count of a file.
<b>uucheck</b>	1137	Checks for files and directories required by BNU.
<b>uucheck</b>	1137	Checks for files and directories required by BNU.
<b>uucico</b>	1139	File transport program for the BNU facility.
<b>uucleanup</b>	1141	Deletes selected files older than a specified number of hours from the BNU spool directory or a named directory.
<b>uucp</b>	1144	Copies files from one AIX system to another AIX system.
<b>uulog</b>	1149	Provides information about <b>uucp</b> and <b>uux</b> activities on a system.
<b>uuname</b>	1151	Provides information about other systems accessible to the local system.
<b>uucpadm</b>	1133	Enters basic <b>uucp</b> configuration information.
<b>uusched</b>	1156	Schedules work for the BNU file transport program.
<b>uustat</b>	1158	Reports the status of and provides rudimentary job control for BNU commands.
<b>uuto</b>	1162	Copies public files from one AIX system to another AIX system, with local system control of file access.
<b>uutry,</b> <b>Uutry,</b> <b>uukick</b>	1164	Contacts a remote system with debugging turned on.
<b>uux</b>	1166	Runs a command on another AIX system.
<b>uuxqt</b>	1172	Executes remote command requests.

## Developing Programs

The following groups of commands are for use in programming.

### Programming in Assembler

<b>as</b>	61	Assembles a source file.
<b>sdb</b>	875	Provides a symbolic debugger for C and assembler programs.

### Programming in C

<b>cb</b>	139.	Puts C source code into a form that is easily read.
-----------	------	---

---

<b>cc</b>	140	Compiles C programs.
<b>cflow</b>	154	Generates a C flow graph of external references.
<b>cpp</b>	210	Performs file inclusion and macro substitution on C Language source files.
<b>cxref</b>	279	Creates a C program cross-reference listing.
<b>factor</b>	416	Factors a number.
<b>ipcrm</b>	537	Removes message queue, semaphore set or shared memory identifiers.
<b>m4</b>	603	Preprocesses files, expanding macro definitions.
<b>lex</b>	562	Generates a C Language program that matches patterns for simple lexical analysis of an input stream.
<b>lint</b>	577	Checks C programs for potential problems.
<b>regcmp</b>	820	Compiles patterns.
<b>sdb</b>	875	Provides a symbolic debugger for C and assembler programs.
<b>tic</b>	1067	Translates <b>terminfo</b> files from source to compiled format.
<b>yacc</b>	1237	Generates a LR(1) parsing program from input consisting of a context-free grammar specification.

## Programming in Miscellaneous Languages

<b>bc</b>	97	Provides an interpreter for arbitrary-precision arithmetic language.
<b>bs</b>	118	Compiles and interprets modest-sized programs.
<b>ctags</b>	261	Makes a file of tags to help locate objects in source files.
<b>m4</b>	603	Preprocesses files, expanding macro definitions.
<b>sno</b>	956	Provides a SNOBOL interpreter.

## Programming in Shell

<b>basename</b>	95	Returns the base name of a string parameter.
<b>cron</b>	220	Runs commands automatically.
<b>crontab</b>	222	Submits a schedule of commands to <b>cron</b> .
<b>csh</b>	225	Interprets commands read from a file or entered from the keyboard.
<b>echo</b>	369	Writes its arguments to standard output.
<b>env</b>	393	Sets the environment for execution of a command.
<b>expr</b>	412	Evaluates arguments as expressions.
<b>find</b>	422	Finds files matching expression.
<b>getopt</b>	485	Parses command line flags and parameters.
<b>line</b>	574	Reads one line from the standard input.
<b>nice</b>	699	Runs a command at a different priority.
<b>nohup</b>	707	Runs a command without hangups and quits.
<b>open</b>	728	Opens a virtual terminal.
<b>sh</b>	913	Interprets commands read from a file or entered at the keyboard.

---

<b>sleep</b>	952	Suspends execution for an interval.
<b>tee</b>	1060	Displays the output of a program and copies it into a file.
<b>test</b>	1064	Evaluates conditional expressions.
<b>time</b>	1068	Times the execution of a command.
<b>true</b>	1099	Returns an exit value of zero.
<b>xargs</b>	1232	Constructs argument lists and runs commands.

## Working with Messages

<b>dspcat</b>	357	Displays all or part of a message catalog.
<b>dspmsg</b>	359	Displays a selected message from a message catalog.
<b>gencat</b>	470	Creates and modifies a message catalog.
<b>gettext</b>	488	Extracts message/insert/help descriptions.
<b>mkcatdefs</b>	651	Preprocesses a message source file.
<b>puttext</b>	796	Updates an output file that contains message/insert/help descriptions.
<b>runcat</b>	852	Pipes data from <b>mkcatdefs</b> to <b>runcat</b> .

## Debugging Programs

<b>crash</b>	215	Examines system images.
<b>dbx</b>	284	Provides a tool to debug and run programs under AIX.
<b>dump</b>	366	Dumps selected parts of an object file.
<b>dumpfmt</b>	368	Formats the VRM dump file.
<b>od</b>	723	Writes the contents of storage to the standard output.
<b>prof</b>	773	Displays program profile data.
<b>adb</b>	33	Provides a general purpose debugger.
<b>sdb</b>	875	Provides a symbolic debugger for C and assembler programs.
<b>time</b>	1068	Times the execution of a command.
<b>timex</b>	1069	Times a command, and reports process data and system activity.
<b>xdbx</b>	1236	Provides an overlaying X-Window application for the dbx symbolic debugger.

## Managing Source Programs Using the Source Code Control System (SCCS)

<b>admin</b>	41	Creates and initializes SCCS files.
<b>cdc</b>	152	Changes the comments in a Source Code Control System (SCCS) delta.
<b>comb</b>	181	Combines SCCS deltas.
<b>delta</b>	310	Creates a delta in a Source Code Control System file.
<b>get</b>	477	Creates a specified version of a Source Code Control System (SCCS) file.
<b>help</b>	513	Provides information about a Source Code Control System (SCCS) message or command or about certain non-SCCS commands.

---

<b>prs</b>	781	Displays a Source Code Control System (SCCS) file.
<b>rmdel</b>	837	Removes a delta from a Source Code Control System (SCCS) file.
<b>sact</b>	862	Displays current Source Code Control System (SCCS) file editing status.
<b>sccsdiff</b>	874	Compares two versions of a Source Code Control System (SCCS) file.
<b>unget</b>	1116	Cancels a previous <b>get</b> command.
<b>val</b>	1175	Validates Source Code Control System (SCCS) files.
<b>what</b>	1213	Displays identifying information in files.

## Managing Object Files

<b>ar</b>	55	Maintains portable libraries used by the linkage editor.
<b>as</b>	61	Assembles a source file.
<b>dump</b>	366	Dumps selected parts of an object file.
<b>ld</b>	557	Links object files.
<b>lorder</b>	591	Finds the best order for member files in an object library.
<b>make</b>	625	Maintains up-to-date versions of programs.
<b>nm</b>	705	Displays the symbol table of an object file.
<b>prof</b>	773	Displays program profile data.
<b>size</b>	949	Displays the section sizes of common object files.
<b>strip</b>	1017	Removes symbol and line number information from a common object file.
<b>touch</b>	1077	Updates the access and modification times of a file.
<b>tsort</b>	1102	Sorts an unordered list of ordered pairs (a topological sort).

## Playing Games

The following groups of commands allow you to play games on the system.

<b>arithmetic</b>	59	Tests arithmetic skills.
<b>back</b>	87	Plays backgammon.
<b>bj</b>	117	Plays blackjack.
<b>craps</b>	214	Plays craps.
<b>fish</b>	427	Plays the card game Go Fish.
<b>fortune</b>	437	Tells a fortune.
<b>hangman</b>	512	Plays hangman, the word-guessing game.
<b>moo</b>	668	Plays a number-guessing game.
<b>number</b>	721	Displays the written form of a number.
<b>quiz</b>	803	Tests your knowledge.
<b>ttt</b>	1104	Plays tic-tac-toe.
<b>turnon</b>	1107	Turns on execute permission for games.
<b>wump</b>	1231	Plays the game Hunt the Wumpus.

## Special Characters

\$- 925  
\$! 924  
\$\$ 924  
\$? 924  
\$# 924

## A

abs command 988  
access times of a file, changing 1077  
accounting  
  ASCII format 29  
  ASCII summary format 19  
  billing summary file 848  
  binary summary format 19  
  combining total accounting files 28  
  connect 24  
  daily 848  
  disk 14  
  line-usage summary 25  
  login 15  
  merging total accounting files 28  
  monthly reports 15  
  process 14, 30  
  reports 15  
  session 15  
  shell procedures 13  
  start 16  
  turn off process 16  
  usage summaries 18  
accounting commands  
  acctcms 18  
  acctcom 20  
  acctcon1 24  
  acctcon2 25

acctdisk 26  
acctdusg 26  
acctmerg 28  
accton 31  
acctprc1 30  
acctprc2 31  
acctwtmp 458  
chargefee 14  
ckpacct 14  
dodisk 14  
fwtmp 457  
lastlogin 15  
monacct 15  
nulladm 15  
prctmp 15  
prdaily 15  
prtacct 16  
runacct 848  
shutacct 16  
startup 16  
turnacct 16  
wtmpfix 458  
accounting file 15  
accounting files  
  /usr/adm/acct/fiscal 15  
  /usr/adm/acct/nite/active 848  
  /usr/adm/acct/nite/ctmp 15  
  /usr/adm/acct/nite/lastdate 848  
  /usr/adm/acct/nite/lock 848  
  /usr/adm/acct/nite/lock1 848  
  /usr/adm/acct/nite/statefile 848  
  /usr/adm/acct/sum 15  
  /usr/adm/acct/sum/loginlog 15  
  /usr/adm/acct/sum/rprt 15  
  /usr/adm/fee 14  
  /usr/adm/pacct 14  
    acctcom 20  
    accton 31  
    ckpacct 14  
    turnacct 16

---

/usr/adm/wtmp  
  acctcon1 24  
  billing summary file 848  
  creating 15  
accounting records  
  ASCII 30  
  ASCII format 24  
  converting ASCII to binary 457  
  converting binary to ASCII 457  
  display 20  
  examining connect records 457  
  login session 24  
  repairing wtmp records 458  
  session 24  
  total accounting login session 25  
accounting report, process 21  
accounting, disk usage 26  
acctcms command 18-19  
acctcom command 20-23  
acctcon1 command 24-25  
acctcon2 command 25  
acctdisk command 26  
acctdusg command 26-27  
acctmerg command 28-29  
accton command 31  
acctprec2 command 31  
acctwtmp command 458  
activity graph, system 865  
activity manager 32  
activity reporter, system 863, 867  
actman command 32  
adb command 33-40  
adding  
  audit bin file 69  
  devices 315  
  groups 1129  
  header flags, SCCS 45  
  users 1129  
  users, SCCS 45  
adduser command 1129-1132  
admin command 41-47  
Advanced Floating-Point Accelerator 143  
Advanced Processor Card 143  
af command 989  
AIX device driver 115  
AIX-KJ commands  
  awk command 83, 84  
  bs command 125  
  cal command 132  
  character class expressions 413, 423, 502,  
  916  
  cpio command 206  
  csh command 225, 232  
  ctab command 257, 260  
  cut command 270  
  dd command 302  
  delta command 312  
  df command 319  
  ed command 375  
  expr command 413  
  file command 420  
  find command 423  
  getty command 491  
  grep command 502, 504  
  lex command 564  
  mkfs command 660  
  od command 723  
  pattern matching 375, 564  
  sh command 916  
  sort command 959  
  tail command 1044  
  tar command 1048  
  tr command 1083  
  translating characters 1083  
  uname command 1114  
  xargs command 1233  
ali command 48-49  
anno command 50-52  
ap command 53-54  
ar command 55-58  
arbitrary precision arithmetic 97  
arithmetic game 59-60  
arithmetic, shell variable 412  
as command 61-62, 143  
assembler 61  
assembling  
  source code  
    as 61  
    asm 61  
    cc 141  
    masm 61  
at command 63-66

---

audit command 67-68, 77  
auditappend command 69-70, 79  
auditbin command 71  
auditd command 72  
auditing commands  
    audit 67  
    auditapp 69  
    auditbin 71  
    auditpr 73  
    auditselect 76  
    auditstream 78  
    auditwrite 80  
auditpr command 73-75  
auditselect command 76  
auditstream command 78  
auditwrite command 80  
awk command 81-86

## B

back game 87  
backing up files 88  
backup command 88-93  
banner command 94  
bar command 1009  
basename command 95-96  
batch command 63-66  
bc command 97-101  
bdiff command 102-103  
bel command 509  
bellmail command 104  
belonging to different groups 506  
bffcreate command 108-109  
bfs command 110-113  
billing summary file, accounting 848  
biod command 114  
biodd\_cfg command 115-116  
bj game 117  
blackjack game 117  
block count of a file, display 1029  
block I/O device driver 115  
branching from nonleaf deltas 481  
break command 931  
bs command 118-128

bucket command 1001  
burst command 129-131

## C

C Language programming  
    See also managing programs  
    See also programming  
    assembling source code 141  
    commands  
        ar 55  
        as 61  
        cb 139  
        cc 140  
        cflow 154  
        cpp 210  
        fcc 141  
        lint 577  
        vcc 141  
        vrmfmt 141  
    cross-reference listing 279  
    files  
        a.out 141  
        formatting source code 139  
        linking object files 141  
        maintaining linkage libraries 55  
        preprocessing source code 141  
        syntax checking 577  
cal command 132-133  
calculating  
    CPU factor 22  
    CPU time 21  
    hog factor 22  
calculator program 97  
calculator, desk 295  
calendar command 134-136  
calprog program 136  
case command 930  
case in Japanese Language Support 257  
cat command 137-138  
cb command 139  
cc command 140-149, 173  
cd command 150-151, 931  
cdc command 152-153

---

CDPATH 922  
 ceil command 991  
 cflow command 154-155  
 changing  
   ASCII accounting records to binary 457  
   binary accounting records to ASCII 457  
   changing permission codes 160  
   current directory 150  
   devices 315  
   files, SCCS 310  
   format of a file 686  
   group identification 689  
   group ownership 156  
   groups 1129  
   login environment 689  
   owner-ID of files or directories 169  
   password 735  
   primary group 689  
   root directory 172  
   SCCS delta comments 152  
   system parameters 171  
   users 1129  
 changing Distributed Services ipc queues  
   table 544  
 changing Distributed Services network  
   Users/Groups table 1109  
 changing Distributed Services node table 685  
 changing password table 801  
 changing state values 361  
 character class expressions (Japanese Language  
   Support) 84, 125, 206, 257, 413, 423, 502, 916  
   equivalence classes  
     ctab command 257  
 character classes 4  
 chargefee command 14  
 charting  
   external references 154  
 checkcw command 277-278  
 checkeq command 395-396  
 checking process accounting files 14  
 checking programs  
   awk command 81  
   bs 118  
   cvid 272  
   managing programs  
     awk 81  
     make 625  
     messages  
     miscellaneous languages  
       ar 55  
       awk 81  
   checkmm command 663  
   checksum of a file, display 1029  
   chgrp command 156-157  
     changing  
       group ownership 156  
   chkcomp command 158-159  
   chmod command 160-163  
   chng command 168  
   chngstate command 164  
   chown command 169-170  
   chparm command 171  
   chroot command 172-173  
   chtcb command 174  
   ckpacct command 14  
   ckprereq command 533  
   clearing an i-node 175  
   clri command 175-176  
   cmp command 177-178  
   col command 179-180  
   collating sequence  
     collating sequence  
       csh command 231  
       ctab command 257  
       sh command 916  
       sort command 958  
     csh command 231  
     ctab command 257  
     li command 568  
     ls command 596  
     NLCTAB environment variable 923  
     sh command 916  
     sort command 958  
   colors  
     setting active display palette 332  
     setting background display 332  
     setting foreground display 332  
   comb command 181-182  
   combining  
     deltas, SCCS 181  
     total accounting files 28  
   comm command 183-184

---

command execution environment 393  
 command line flag parsing 485  
 command usage summary 18  
 commands  
   See accounting commands  
   See AIX-KJ commands  
   See auditing commands  
   See C Language programming  
   See communication commands  
   See controlled access mode commands  
   See editors  
   See filter commands  
   See graphics commands  
   See maintenance commands  
   See managing programs, commands  
   See Multi-User Services commands  
   See programming  
   See reading standard input  
   See system group commands  
   See text processing commands  
   See writing to standard output  
 comments in SCCS files, including kanji characters 312  
 communication commands  
   confer 189  
   connect 198  
   ct 254  
   cu 263  
   Cvt 274  
   edconfig 385  
   fmt 428  
   mail, Mail 608  
   mailstats 623  
   mesg 642  
   news 691  
   rmail 836  
   sendmail 897  
   uucheck 1137  
   uucico 1139  
   uucleanup 1141  
   uucp 1144  
   uulog 1149  
   uname 1151  
   uupick 1153  
   uusched 1156  
   uustat 1158  
   uuto 1162  
   uutry 1164  
   uux 1166  
   uuxqt 1172  
   wall 1208  
   who 1219  
   300 1262  
   4014 1264  
   450 1265  
 communication, interprocess status 539  
 comp command 185-188  
 comparing  
   directories  
     dircmp 328  
   files 883  
     bdiff 102  
     cmp 177  
     diff 320  
     diffmk 326  
     diff3 323  
     dircmp 328  
   SCCS files 874  
 compilers  
   bs 118  
   cc 140  
   sno 956  
 compress program 971  
 compressing files 730  
 concatenate files 137  
 concurrent groups 506  
 conditional expressions, evaluating 1064  
 confer command 189-193  
 config command 194-195  
 configuration information 194  
 configuring  
   block I/O device driver 115  
 conflict command 196-197  
 connect accounting 24  
 connect command 198-201  
 consistency check and repair of files  
   dfscck command 447  
   fsck command 445  
 constant-width text 275  
 constructing a file system 658  
 contents of directory, listing 346, 567  
 context split 252

---

continue command 931  
 controlled access mode commands  
   audit 67  
   auditapp 69  
   auditbin 71  
   auditpr 73  
   auditselect 76  
   auditstream 78  
   auditwrite 80  
   chtcb 174  
   getty 491  
   login 584  
   logout 590  
   tsh 1100  
   watch 1209  
 conversion routines  
   fromnls 302  
   fromsjis (Japanese Language Support) 302  
   tonls 302  
   tosjis (Japanese Language Support) 302  
 converting  
   ASCII accounting records to binary 457  
   binary accounting records to ASCII 457  
   kanji characters 302  
 copy command 202-204  
 copying  
   AIX files  
     copy 202  
     cp 202  
   DOS files  
     dosread 348  
     doswrite 350  
 cor command 1002  
 cp command 202-204  
 cpio command 205-209  
 cpp command 143, 210-213  
 CPU factor computation 22  
 CPU time computation 21  
 craps game 214  
 crash command 215-219  
 creating  
   C program cross-reference listing 279  
   delta, SCCS 310  
   mount table 911  
   SCCS files 41  
   special file 661  
     specified version of an SCCS file 477  
 cron command 220-221, 848  
   used with the sa1 command 864  
   used with the sa2 command 864  
 crontab command 222-224  
 cross-reference listing, C program 279  
 csh command 225-251  
 csplit command 252-253  
 ct command 254-256  
 ctab command 257-260  
 ctags command 261-262  
 cu command 263-268  
 current directory  
   current directory, changing 150  
 cusum command 991  
 cut command 269-271  
 cvid command 272-273  
 cvrtopt command 509  
 Cvt command 274  
 cw command 275-278  
   used in pipeline with nroff 710  
 cxref command 279-280

## D

daemon, error-logging 398  
 daemon, error-logging termination 404  
 daemon, terminal logging 1072  
 daemon, tlogger 1071  
 daily accounting 848  
 database operator 547  
 date command 281-283, 356, 363  
 dbx 1236  
 dbx command 284-294  
 dc command 295-298  
 dcopy command 299-300  
 dd command 301-305  
 debugger, file system 450  
 defining shell functions 931  
 defkey command 306-307  
 del command 308-309  
 deleting  
   delta from SCCS file 837  
 devices 315

---

- directories
  - rm 833
  - rmdir 838
- DOS files 345
- files
  - del 308
  - rm 833
  - skulker 951
- groups 1129
- repeated words 1118
- users 1129
- users, SCCS 45
- delta command 310-312
- delta summary of SCCS file 482
- deltas, branching from nonleaf 481
- deroff command 313-314
- description file, make command 630
- desk calculator 295
- device (special) files
  - /dev/audit 78
  - /dev/null
    - acctcom 20
    - standard input assigned to 20
- adding 315
- changing 315
- creating 661
- deleting 315
- device driver
  - block I/O 115
- device name 316
- devices
  - devices command 315
- devnm command 316-317
- df command 318
- dfsck command 447-449
- di command 567-573
- diagnostic
  - token-ring 1097
  - trdiag 1097
- diff command 177, 320-322
- diffmk command 326-327
- diff3 command 323-325
- dircmp command 328-329
- directories
  - changing
    - owner-ID 169
  - comparing
    - dircmp 328
  - listing contents
    - di 567
    - DOS directories 346
    - li 567
    - ls 595
  - removing
    - rm 833
    - rmdir 838
- directory 682
  - change root 172
  - changing
  - changing current 150
  - create 657
  - moving 682
  - renaming 682
  - return path name 95
- directory contents, listing 346, 567
- dirname command 95-96
- disk usage accounting 26
- disk usage summary 364
- diskusg command 330-331
- display command 332-335
- display station
  - changing DMA pinned page 333
  - setting active color palette 332
  - setting background colors 332
  - setting fonts 332
  - setting foreground colors 332
- displaying
  - a calendar 132
  - accounting report 15
  - audit record
    - auditpr 73
    - auditselect 76
  - audit trail 73
  - compressed files 730
  - connect accounting records 457
  - contents of i-nodes 545
  - corresponding group names and IDs 517
  - corresponding user names and IDs 517
  - current directory 800
  - date 281
  - documents formatted with the Memorandum
    - Macros 663

---

file checksum 1029  
files 110, 137  
formatted files 744, 761  
login name 589  
news items 691  
packed files 730  
process accounting records 20  
process status 786  
profile data 773  
SCCS file editing activity 862  
session record 15  
squeezed files 730  
system images 215  
system parameters 171  
total accounting report 16  
dist command 336-339  
Distributed Services  
  dsldxprof 355  
dividing a file into pieces 974  
DMA channel, setting 910  
dodisk command 14  
dos command 341-344  
dosdel command 345  
dosdir command 346-347  
dosread command 348-349  
doswrite 350-351  
dp command 352-353  
drill in arithmetic skills 59  
dsipc command 354  
dsldxprof command 355  
dspcat command 357-358  
dspmsg command 359-360  
dsstate command 361-362  
dsxlate command 363  
dtoc command 1074  
du command 364-365  
dump command 366-367  
dump, extracting error records 397  
dump, octal 723  
dumpfmt command 368

## E

echo command 369-370  
ed command 371-384  
edconfig command 385-386  
edit command 387-392  
editors  
  ed 371  
  edit 387  
  ex 407  
  ged 463  
  red 371  
  sed 887  
  tvi 1108  
  vedit 1187  
  vi 1187  
  vview 1187  
egrep command 501-505  
end a process 552  
env command 393-394  
environment, changing login 689  
eqn command 395, 396  
  constructs removed by the deroff  
  command 313  
  used in pipeline with nroff 710  
  used with tbl 1053  
erase command 461  
errdead command 397  
errdemon command 398-399  
error-logging daemon 398  
error-logging daemon termination 404  
error records extraction from dump 397  
error report 400  
errpd command 399, 403  
errpt command 400-403  
errstop command 404  
errupdate 406  
errupdate command 405  
eval command 932  
evaluating expressions  
  expr 412  
  test 1064  
ex command 407-411  
examining  
  connect accounting records 457

---

- contents of i-nodes 545
- files 110
- system images 215
- system parameters 171
- exec command 932
- exercising link system call 575
- exit command 932
- exp command 992
- expanding packed files 730
- export command 932
- expr command 412-415
- expression evaluation 412
- extended character support
  - See international character support
- external references, flow graph 154
- extract error records from dump 397

## F

- factor command 416
- factoring a number 416
- false command 1099
- fcc command 141
- ff command 417-419
- fgrep command 501-505
- file
  - display checksum 1029
- file command 420-421
- file formats
  - acct 18, 30
  - ar 55
  - backup 88
  - tacct 28
  - utmp 458
  - wtmp 458
- file pattern search 501
- file system
  - See also device (special) files
  - See also maintenance commands
  - See also system files
  - backing up 88
  - make available for use 669
  - make unavailable for use 1112
  - making 658

- moving a directory 682
- renaming a directory 682
- unmount 1112
- file system debugger 450
- files
  - See also accounting files
  - See also device (special) files
  - See also managing programs, files
  - See also system files
  - a.out 141
  - auditing files
  - backing up 88
  - calendar 134
  - changing
    - owner-ID 169
  - checking consistency
    - dfscck command 447
    - fsck command 445
  - comparing 883
    - cmp 177
    - diff 320
    - difffmk 326
    - diff3 323
    - dircmp 328
  - comparing large files 102
  - compressing 730
  - concatenating 137
  - copying
    - AIX files 202
    - DOS files 348, 350
  - creating SCCS files 41
  - deleting
    - del 308
    - DOS files 345
  - determining type 420
  - displaying 137
  - displaying formatted files 744, 761
  - expanding 730
  - finding 422
  - identifying the processes using a file 455
  - initializing SCCS files 41
  - linking 581
  - merge lines 736
  - merging 958
  - modifying the user mask 1110
  - naming SCCS files 43

---

packing 730  
 parallel merging 736  
 removing  
   rm 833  
   skulker 951  
 repairing  
   dfscck command 447  
   fsck command 445  
 repairing damage 215  
 return base name 95  
 scanning 110  
 searching 110  
 searching for a pattern 501  
 serial merging 736  
 setting file-creation permission code  
   mask 1110  
 sorting 958  
 squeezing 730  
 text  
   changing the format 686  
 transforming 301  
 translating 301  
 unpacking 730  
 unsqueezing 730  
 writing the last part 1044  
 3-way comparison 323  
 filter commands  
   acctcom 20  
   acctcon1 24  
   acctmerg 28  
   awk 81  
   bdiff 102  
   cb 139  
   cmp 177  
   col 179  
   comb 181  
   cw 275  
   definition of 914  
   fwtmp 457  
   hp 514  
   nl 701  
   nroff 710  
   paste 736  
   ptx 794  
   tbl 1053  
   troff 710  
   wtmpfix 458  
 find command 422, 426  
   acctdusg 26  
 find hyphenated words 516  
 find necessary order of files in an object  
   library 591  
 fish game 427  
 fixed minidisk information 650  
 Floating-Point Accelerator 143  
 floating point configuration 444  
 floor command 992  
 flow graph of external references 154  
 fmt command 428  
 folder command 429-432  
 folders command 433-435  
 fonts  
   setting virtual terminal 332  
 for command 930  
 format command 436  
 formats  
   See file formats  
 formatting C Language source code 139  
 formatting text  
   constant-width text 275  
   for a phototypesetter 709  
   for a printing device 709  
   inverse line feeds and half-line feeds 179  
   mathematical text 395  
   tables for nroff, troff 1053  
 fortune game 437  
 forw command 438-443  
 forwarding mail 105  
 fptype command 444  
 free disk space, reporting 318  
 fromnls conversion routine 302, 303  
 fromsjis conversion routine (Japanese Language  
   Support) 302, 303  
 fsck command 445-449  
 fsdb command 175, 450-454  
 fuser command 455-456  
 fwtmp command 457

---

**G**

games

- arithmetic 59
- back 87
- backgammon 87
- bj 117
- blackjack 117
- craps 214
- fish 427
- fortune 437
- hangman 512
- man 635
- moo 668
- number 721
- quiz 803
- ttt 1104
- wump 1231

gamma command 993

gd command 510

gdev commands 460-462

ged command 463-469

gencat command 470-474

gend command 475-476

generating C program cross-reference listing 279

generating names from i-numbers 683

get command 310, 311, 477-484

getopt command 485-487

gettext command 488-489

getty command 490-493

going to maintenance mode 946

graph command 494-496

graph, system activity 865

graphical editor 463

graphics command 497-498

graphics commands

- abs 988
- af 989
- bar 1009
- bel 509
- bucket 1001
- ceil 991
- cor 1002
- cusum 991
- cvrtopt 509
- dtoc 1074
- erase 461
- exp 992
- floor 992
- gamma 993
- gd 510
- ged 463
- gend 475
- graph 494
- graphics 497
- gtop 510
- hardcopy 461
- hilo 1003
- hist 1010
- hpd 460
- label 1011
- list 993
- log 994
- lreg 1004
- mean 1005
- mod 995
- pair 996
- pd 510
- pie 1012
- plot 1014
- point 1006
- power 997
- prime 986
- prod 1006
- ptog 510
- qsort 1007
- quit 510
- rand 987
- rank 1007
- remcom 510
- root 997
- round 998
- siline 999
- sin 1000
- spline 972
- subset 1000
- td 461
- tekset 461
- title 1016
- total 1008

---

tplot 1079  
ttoc 1075  
utility commands 508  
var 1008  
vtoc 1075  
whatis 511  
graphing  
  external references 154  
greek command 499-500  
grep command 501-505  
group  
  adding 1129  
  changing 1129  
  deleting 1129  
group identification, changing 689  
group IDs and names, displaying 517  
group membership 506  
group membership, display 506  
groups command 506-507  
gtop command 510  
guess a word 512

## H

hangman game 512  
hardcopy command 461  
hash command 932  
hashcheck command 970  
hashmake command 970  
header flags, SCCS 43, 45  
help command 513  
hilo command 1003  
hist command 1010  
hog factor computation 22  
HOME 922  
hp command 514-515  
hpd command 460-461  
hyphen command 516  
hyphenated words 516

## I

i-node content, displaying 545  
i-node examination 545  
i-numbers  
  generating names 683  
I/O counts 22  
id command 517  
if command 930  
IFS 924  
inc command 518-520  
init command 521-523  
initialization, normal startup 806  
initializing  
  SCCS files 41  
install command 524-526  
install-mh command 527-528  
installp command 529-536  
interactive processor 97  
international character support  
  at command 64  
  batch command 64  
  collating sequence  
  csh command 231  
  ctab command 257  
  dd command 302  
  dfsck command 447  
  equivalence classes  
  fsck command 447  
  fsdb command 452  
  li command 568  
  ls command 596  
  od command 724  
  print command 770  
  ps command 787  
  sh command 923  
  sort command 958  
  stty command 1023  
interpolating a smooth curve 972  
interpreters  
  bc 97  
  bs 118  
  sno 956  
Interprocess Communication key mapping  
  installation 354

---

interprocess communication status 539  
inudocm command 1125-1126  
inuupdt command 1127  
ipc queues table  
    access 544  
ipcrm command 537-538  
ipcs command 539-543  
ipctable command 544  
istat command 545-546

## J

join command 547-550  
joinconf command 189  
joining database files 547  
joint editing of an SCCS file 482

## K

kanji characters, used in variable names 83  
kcore minutes, definition 19  
keyboard command 551  
keyboard, redefine 306  
kill all nonancestral processes 555  
kill command 552-554  
killall command 555

## L

label command 1011  
language support  
    See international character support  
lastlogin command 15  
ld command 143, 557-561  
lex command 562-566  
li command 567-573, 838  
LIBPATH 922  
library maintainer 55

library search order 144  
line command 574  
line editor 371  
line numbering filter 701  
line printer back end 593  
link command 575-576  
link library maintaining 55  
linkage editor 557  
linking  
    files 581  
    object files  
        cc 141  
        ld 557  
lint command 577-580  
list command 993  
listing  
    directory contents  
        di 567  
        li 567  
        ls 595  
    DOS directory contents 346  
    file names for a file system 417  
    statistics for a file system 417  
ln command 581-582  
locator command 583  
log command 994  
logged error report 400  
login command 584-586  
login environment, changing 689  
login names 491  
login session records 24  
loginx command 587-588  
LOGNAME 922  
logname command 589  
logout command 590  
looking at  
    connect accounting records 457  
    contents of i-nodes 545  
    files 110  
    system images 215  
    system parameters 171  
lorder command 591-592  
lp command 593-594  
lreg command 1004  
ls command 595-599

---

**M**

- m4 command 603, 607
- macroprocessor 603
- MAIL 922
- mail command 107
- mail, Mail command 608-622
- MAILCHECK 922
- MAILMSG 923
- MAILPATH 922
- mailstats command 623-624
- maintaining groups of programs 625
- maintaining linkage libraries 55
- maintenance commands
  - backup 88
  - clri 175
  - cpio 205
  - mount 669
  - umount 1112
  - unmount 1112
- maintenance mode, going to 946
- make a directory 657
- make a tags file 261
- make command 625-631
- makekey command 634
- making a file system 658
- making two files the same 102
- man pubs 635-636
- manager, virtual terminals 32
- managing programs
  - See also programming
  - See also SCCS
  - commands
    - admin 41
    - cdc 152
    - comb 181
    - delta 310
    - get 477
    - help 513
    - prs 781
    - sact 862
    - sccsdiff 874
    - unget 1116
    - val 1175
    - what 1213
- files
  - auxiliary files 478
  - creating a delta 310
  - g-file 43, 310, 311, 477, 478, 483
  - l-file 478, 482
  - lock file 43
  - p-file 478, 479, 482
  - s-file 478
  - x-file 43
  - z-file 43, 478
- make 625
- security management
  - auditbin command 71
  - auditstream command 78
- z-file 479
- mant command 666
- mark command 637-639
- marking differences between files 326
- mdrc command 640-641
- mean command 1005
- membership, display group 506
- merging files 958
- merging lines in files 736
- merging total accounting files 28
- mesg command 642
- Message Handling commands
  - ali 48
  - anno 50
  - ap 53
  - burst 129
  - comp 185
  - conflict 196
  - dist 336
  - dp 352
  - folder 429
  - folders 433
  - forw 438
  - inc 518
  - install-mh 527
  - mark 637
  - mhl 643
  - mhmail 646
  - mhpath 648
  - msgchk 675
  - msh 677
  - next 694

---

packf 733  
 pick 748  
 post 758  
 prev 765  
 prompter 778  
 rcvdist 808  
 rcvpack 810  
 rcvstore 812  
 rcvtty 815  
 refile 817  
 repl 821  
 rmf 839  
 rmm 841  
 scan 871  
 send 893  
 show 942  
 slocal 954  
 sortm 965  
 spost 978  
 vmh 1203  
 whatnow 1215  
 whom 1222  
 message queue removal 537  
 messages, permitting 642  
 messages, refusing 642  
 messages, send 104  
 messages, sending 1225  
 MH commands  
   See Message Handling commands  
 mhl command 643-645  
 mhmail command 646-647  
 mhpath command 648-649  
 minidisks 650  
 minidisks command 650  
 mkcatdefs command 651-656  
 mkdir command 657  
 mkfs command 658-660  
 mknod command 661-662  
 mm command 663-665  
 mmt command 666-667  
 mod command 995  
 modification request number 311  
 modification times of a file, changing 1077  
 modifying  
   changing permission codes 160  
   current directory 150  
   devices 315  
   files, SCCS 310  
   group identification 689  
   group ownership 156  
   groups 1129  
   login environment 689  
   owner-ID of files or directories 169  
   password 735  
   primary group 689  
   root directory 172  
   SCCS delta comments 152  
   system parameters 171  
   users 1129  
   modifying access times of a file 1077  
   modifying modification times of a file 1077  
   monacct command 15  
   moo game 668  
   mount a file system 669  
   mount command 669-673  
   mount table, creating 911  
   mountd command 674  
   move command 679-681  
   moving a directory 682  
   moving files 679  
   msgchk command 675-676  
   msh command 677-678  
   mt command 666  
 Multi-User Services commands  
   acctcms 18  
   acctcom 20  
   acctcon1 24  
   acctcon2 25  
   acctdisk 26  
   acctdusg 26  
   acctmerg 28  
   accton 31  
   acctprc1 30  
   acctprc2 31  
   acctwtmp 458  
   chargefee 14  
   ckpacct 14  
   dodisk 14  
   fwtmp 457  
   id 517  
   lastlogin 15  
   mesg 642

---

monacct 15  
nulladm 15  
prctmp 15  
prdaily 15  
prtacct 16  
runacct 848  
sar 867  
shutacct 16  
startup 16  
turnacct 16  
who 1219  
wtmpfix 458  
300 1262  
4014 1264  
450 1265  
mv command 679-681  
mmdir command 682  
mvmd command 534-535  
mvt command 666

## N

ncheck command 683-684  
ndtable command 685  
neqn command 395-396  
    used with tbl 1053  
network Users/Groups table  
    access 1109  
newform command 686-688  
newgrp command 689-690, 933  
news command 691-693  
next command 694-695  
nice command 699-700  
nl command 701-704  
NLCTAB 923  
nm command 705-706  
node table  
    access 685, 801  
nohup command 707-708  
normal startup initialization 806  
nroff command 709-720  
    tbl, preprocessor 1053  
nulladm command 15  
number factoring 416

number game 721  
numbering lines 701

## O

object library  
    ordering relation 591  
octal dump 723  
od command 723-725  
open command 728, 729  
    used after actman command 32

## P

pack command 730-732  
packf command 733-734  
pair command 996  
parallel merging of lines in files 736  
parameters  
    work station  
        erase 490  
        kill 490  
        logmodes 490  
        owner 492  
        parity 490  
        program 492  
        protection 492  
        runmodes 490, 491, 492  
        special purpose options 492  
        speed 490  
parameters, setting work station 1018  
    terminal mapping 1023  
parsing command line flags 485  
passwd command 735  
password table 801  
password, change 735  
passwords, characteristics  
paste command 736-738  
PATH 923  
path name, return directory 95  
pattern matching

---

acctcom 22  
awk 81  
pattern matching (Japanese Language Support) 232  
pattern, search for 501  
pcat command 730-732  
pd command 510  
pdelay command 791-793  
pdisable command 741-743  
penable command 791-793  
perform disk accounting functions 14  
perform monthly accounting 15  
performing process accounting 30  
permission codes, changing 160  
    changing  
        permissions 160  
    permitting messages 642  
pg command 744-747  
phold command 741-743  
pick command 748-752  
pie command 1012  
piobe command 753-756  
pipe fitting 1060  
pipeline  
    asynchronous execution 914  
    conditional execution 914  
    definition of 914  
    sequential execution 914  
plot command 1014  
point command 1006  
port characteristics 490  
port characteristics, setting 490  
portmap 757  
post command 758-760  
power command 997  
pr command 761-764  
prctmp command 15  
prdaily command 15, 848  
precision arithmetic 97  
preprocessing  
    source code  
        cc 141  
        cpp 210  
preprocessor  
    macro 603  
prev command 765-766

primary group 506  
primary group, changing 689  
prime command 986  
print command 767-772  
printer back end 593  
printing  
    a calendar 132  
    accounting report 15  
    audit record  
        auditpr 73  
        auditselect 76  
    audit trail 73  
    compressed files 730  
    corresponding group names and IDs 517  
    corresponding user names and IDs 517  
    current directory 800  
    date 281  
    documents formatted with the Memorandum Macros 663  
    file checksum 1029  
    formatted files 744, 761  
    login name 589  
    news items 691  
    packed files 730  
    process accounting records 20  
    process status 786  
    profile data 773  
    SCCS file editing activity 862  
    session record 15  
    squeezed files 730  
    total accounting report 16  
priority, running a command 699  
process a report of logged errors 400  
process accounting 30  
process accounting records, display 20  
process accounting report 21  
process accounting, turn off 16  
    /usr/adm/wtmp  
        shutacct 16  
process suspension 552  
prod command 1006  
producing C program cross-reference listing 279  
prof command 144, 773-774  
profiler commands 775  
profiling the operating system 775

---

program checking, C programs 577  
program maintenance 625  
program update 1127  
program updating 625  
programming  
  See also managing programs  
  assembler  
    as 61  
  assembling source code 141  
  C Language  
    cb 139  
    cc 140  
    cflow 154  
    cpp 210  
    fcc 141  
    formatting source code 139  
    lint 577  
    vcc 141  
    vrmfmt 141  
  debugging programs  
  files  
    a.out 141  
  linking object files 141  
  preprocessing source code 141  
  the shell  
programs, checking  
  awk command 81  
  bs 118  
  cvid 272  
  managing programs  
    awk 81  
    make 625  
  messages  
  miscellaneous languages  
    ar 55  
    awk 81  
prompter command 778-779  
proto command 780  
prs command 781-785  
prtacct command 16  
ps command 786-790  
pshare command 791-793  
pstart command 791-793  
PS1 923  
PS2 923  
ptecms.awk program 17

ptelus.awk program 17  
ptog command 510  
ptx command 794-795  
puttext command 796-797  
pwck command 798-799  
pwd command 800, 933  
pwtable command 801

## Q

qdaemon command 802  
qsort command 1007  
query terminal characteristics 1062  
quit command 510  
quiz game 803-805

## R

rand command 987  
rank command 1007  
rc command 806, 807  
  startup 16  
rcvdist command 808-809  
rcvpack command 810-811  
rcvstore command 812-814  
rcvtty command 815-816  
read command 933  
read operations 22  
reading one line 574  
reading standard input  
  acctcon1 24  
  acctmerg 28  
  as 61  
  at 63  
  awk 81  
  batch 63  
  bdiff 102  
  cb 139  
  cmp 177  
  comb 181  
  fwtmp 457

---

wtmpfix 458  
readonly command 933  
recovering from a system crash 215  
red command 371  
redefine keyboard 306  
refile command 817-819  
refusing messages 642  
regcmp command 820  
rejecting lines common to two sorted files 183  
relational database operator 547  
remcom command 510  
reminder service 134  
remote system mail 836  
remote system, connection 198  
remotely-settable hardware tabs 1041  
remove a message queue 537  
remove a semaphore set 537  
remove a shared memory id 537  
remove command 16  
removing  
    delta from SCCS file 837  
    devices 315  
    directories  
        rm 833  
        rmdir 838  
    DOS files 345  
    files  
        del 308  
        rm 833  
        skulker 951  
    groups 1129  
    repeated words 1118  
    users 1129  
    users, SCCS 45  
renaming a directory 682  
renaming files 679  
repairing damaged files 215  
repairing wtmp records 458  
repeated words, deleting 1118  
repl command 821-825  
report process data and system activity 1069  
report, process accounting 21  
reporting free disk space 318  
reports  
    SCCS 182  
reports, accounting

restarting tlogger daemon 1071  
restore command 826-831  
return base file name or directory path  
    name 95  
return command 933  
returning a true or false value 1099  
rexd command 832  
rm command 833-835, 838  
rmail command 836  
rmdel command 837  
rmdir command 838  
rmf command 839-840  
rmm command 841-842  
root command 997  
root directory, changing 172  
round command 998  
rpcgen command 843  
rpcinfo command 845  
rsh command 935  
rsprayd command 983  
rstatd command 847  
runacct command 848-851  
runcat command 852-853  
running a command at low priority 699  
running commands at a later time 63  
rup command 854  
rusers command 856  
rusersd command 858  
rwall command 859  
rwalld command 861

## S

sact command 862  
sadc command 863-864  
sag command 865-866  
sar command 867-870  
saving files 88  
sa1 command 863-864  
sa2 command 863-864  
scan command 871-873  
scanning files 110  
SCCS

    branching from leaf deltas 481

---

changing delta comments 152  
 checking the structure of SCCS files 45  
 comments, including kanji characters 312  
 creating a file 41  
 creating a specified version of a file 477  
 Data Keywords 781  
 delta summary 482  
 getting help information 513  
 header flags 43  
 identification keywords 480  
 initializing a file 41  
 interpreting errors 513  
 joint editing of files 482  
 modification request (MR) number 46, 311  
 Modification Requests 152  
 naming a file 43  
 recalculating the SCCS file checksum 46  
 removing a delta 837  
 reports 182  
 SID (SCCS Identification) 311  
 specifying version date cutoff 482  
 sccsdiff command 874  
 scheduling commands 63  
 scheduling queue requests 802  
 sdb command 875-882  
 sdiff command 883-884  
 search order, library 144  
 searching files 110  
 secure command 885-886  
 sed command 887-892  
 segment files 252  
 selecting fields from a file 269  
 selecting lines common to two sorted files 183  
 semaphore set removal 537  
 send command 893-896  
 sending messages 104, 1225  
 sendmail command 897-909  
 serial merging of lines in files 736  
 session records 24  
 set command 485, 933  
 set environment 393  
 setdma command 910  
 setmnt command 911-912  
 setting DMA channel 910  
 setting file-creation permission code  
 mask 1110  
 setting port characteristics 490  
 setting tabs on a work station 1041  
 setting the date 281  
 setting the parameters for a work station 1018  
 sh command 913-937  
   See also shell  
   command line separators and  
   terminators 914  
 SHACCT 924  
 shared memory id removal 537  
 shell 924  
   actman 32  
   blank interpretation 930  
   built-in commands 931  
   command environment 919  
   command execution 914  
   command-line substitution 917  
   command substitution 925  
   conditional substitution 920  
   control commands 930  
   diagnostic output 927  
   file descriptors 926  
   file-name substitution 916  
   inline input documents 928  
   positional parameters 918  
   predefined special variables  
   profile file 915  
   quoting mechanisms 925  
   redirection of input and output 926, 930  
   running the shell 935  
   shell variables  
     \$- 925  
     \$! 924  
     \$\$ 924  
     \$? 924  
     \$# 924  
     CDPATH 922  
     HOME 922  
     IFS 924  
     LIBPATH 922  
     LOGNAME 922  
     MAIL 922  
     MAILCHECK 922  
     MAILMSG 923  
     MAILPATH 922  
     NLCTAB 923

---

PATH 923  
 PS1 923  
 PS2 923  
 SHACCT 924  
 SHELL 924  
 TIMEOUT 924  
 signals 915  
 standard input and output 926  
 subshell 915  
 summary of redirection options 929  
 user-defined variables 918  
 variables 917  
 shell command 938  
 shell environment 491  
 shell procedures for accounting 13  
 shell variable arithmetic 412  
 shift command 934  
 shlib command 939-941  
 show command 942-944  
 show login records 15  
 showing  
   a calendar 132  
   accounting report 15  
   audit record  
     auditpr 73  
     auditselect 76  
   audit trail 73  
   compressed files 730  
   corresponding group names and IDs 517  
   corresponding user names and IDs 517  
   current directory 800  
   date 281  
   documents formatted with the Memorandum  
     Macros 663  
   file checksum 1029  
   formatted files 744, 761  
   login name 589  
   news items 691  
   packed files 730  
   process accounting records 20  
   process status 786  
   profile data 773  
   SCCS file editing activity 862  
   session record 15  
   squeezed files 730  
   total accounting report 16  
   showmount command 945  
   shutacct command 16  
   shutdown command 946-948  
   shutting down the system 946  
   siline command 999  
   sin command 1000  
   sininstallck command 1036  
   size command 949-950  
   skulker command 951  
   sleep command 952-953  
   slocal command 954-955  
   sno command 956-957  
   sort command 958-964  
   sorting characters (Japanese Language  
     Support) 959  
   sorting files 958  
   sortm command 965-966  
   sound command 967-968  
   source code  
     external references flow graph 154  
   Source Code Control System  
     See SCCS  
   special (device) files  
     /dev/audit 78  
     /dev/null  
       acctcom 20  
       standard input assigned to 20  
   adding 315  
   changing 315  
   creating 661  
   deleting 315  
   specifying version date cutoff 482  
   spell command 969-971  
   spellin command 970  
   spellprog program 971  
   spline command 972-973  
   split command 974  
   split files by context 252  
   splitting a file into pieces 974  
   splp command 975-977  
   spost command 978-980  
   spray command 981  
   squeezing files 730  
   standard input  
     acctcon1 24  
     acctmerg 28

---

as 61  
at 63  
awk 81  
batch 63  
bdiff 102  
cb 139  
cmp 177  
comb 181  
fwtmp 457  
wtmpfix 458  
standard output  
  acctcon1 24  
  acctmerg 28  
  acctwtmp 458  
  awk 81  
  bdiff 102  
  cal 132  
  cb 139  
  cflow 154  
  cmp 177  
  comb 181  
  fwtmp 457  
  wtmpfix 458  
start accounting functions 16  
starting up the system 521  
startup command 16  
startup initialization 806  
startup shell  
  actman 32  
stat commands 984-1016  
state values  
  changing 361  
status of interprocess communication 539  
stop a process 552  
stopping error-logging daemon 404  
stopping tlogger daemon 1071  
stream editor 887  
strings 94  
strip command 1017  
stty command 1018-1025  
su command 1026-1028  
subset command 1000  
sum command 1029  
summarize disk usage 364  
summary of command usage 18  
superblock update 1030  
superuser authority 770, 771, 1050, 1159  
  biodd\_cfg 115  
  commands  
    adduser 1129  
    at 63  
    chmod 160  
    chroot 172  
    ckprereq 533  
    cpio 206  
    crontab 222  
    cvid 272  
    date 281  
    devices 315  
    dfscck 447  
    errdemon 398  
    errstop 404  
    fsck 446  
    installc 166  
    installp 529  
    killall 555  
    mesg 642  
    minidisks 650  
    mount 669  
    mvmd 534  
    nice 699  
    passwd 735  
    print 770  
    removing scheduled jobs 64  
    rm 833  
    shown in report 21  
    shutdown 946  
    tar 1049  
    tlog 1071  
    updatec 167  
    updatep 1122  
    users 1129  
    uustat 1159  
  dsipc 354  
  dsldxprof 355  
  dsstate 361  
  dsxlate 363  
  ipctable 544  
  ndtable 685  
  pwtable 801  
  removing files 833  
  ugtable 1109

---

suspend a process 552  
 sync command 1030  
 syntax checking, C programs 577  
 sysck command 1031  
 syslogd command 1037-1039  
 system activity graph 865  
 system activity reporter 863, 867  
 system files  
   \$HOME/.profile 1027  
   /dev/null  
     expr 415  
   /etc/.ilog 585, 1220  
   /etc/budate  
     mount 90  
   /etc/environment 523, 585  
   /etc/fileystems 272, 446, 456, 658  
     cvid 272  
     mount 89, 669, 670  
   /etc/group 1129  
     groups 506  
   /etc/magic 420  
   /etc/mnttab  
     mount 669  
     umount 1113  
   /etc/ogroup 1129  
   /etc/opasswd 1129  
   /etc/passwd 585, 1026, 1027, 1129  
     acctdisk 27  
     acctprcl 30  
     groups 506  
   /etc/portstatus 522, 742, 792  
   /etc/profile 1027  
   /etc/qconfig 754  
   /etc/rasconf 398  
   /usr/adm/sulog 1026  
   /usr/adm/user.cfile 1129, 1131  
   /usr/adm/wtmp 458, 1220  
   /usr/games/lib/backrules 87  
   /usr/lib/cron/at.allow 63  
   /usr/lib/cron/at.deny 63  
   /usr/lib/eign 795  
   /usr/lib/terminfo 746  
   /usr/lib/tmac 313  
   a.out 61  
   connect.con 198  
   group 423, 689  
   passwd 423, 689  
   ports 741, 791, 1220  
   portstatus 741, 791  
   utmp 1220  
 system group commands  
   adduser 1129  
   backup 88  
   biodd\_cfg 115  
   ckprereq 533  
   cvid 272  
   date 281  
   devices 315  
   dsipc 354  
   dslxprof 355  
   dsstate 361  
   dsxlate 363  
   installc 166  
   installp 529  
   ipctable 544  
   minidisks 650  
   mount 669  
   mvmd 534  
   ndtable 685  
   print 770  
   pwtable 801  
   turnoff 1107  
   turnon 1107  
   ugtable 1109  
   updatec 167  
   users 1129  
 system image examination 215  
 system parameters, changing or examining 171  
 system procedures  
   restarting the terminal-logging  
     daemon 1071  
   starting the error-logging daemon 398  
   starting the terminal-logging daemon 1072  
   stopping the terminal-logging daemon 1071,  
     1072  
 system startup 521

---

**T**

- tab command 1040
- tabs command 1041-1043
- tail command 1044-1046
- tape archiver 1048
- tapechk command 1047
- tar command 1048-1052
- tbl command 1053, 1055
  - descriptions removed by the deroff command 313
  - used in pipeline with nroff 710
- tc command 1056-1057
- tctl command 1058-1059
- td command 461-462
- tee command 1060-1061
- tekset command 461
- termdef command 1062
- terminal characteristics 1062
- terminal-logging daemon 1072
- terminal mapping 1023
- terminals
  - DASI 300 1262
  - DASI 300s 1262
  - DASI 450 1265
  - Diablo 1620 1265
  - HP2621 514
  - HP2640 514
  - phototypesetter simulator 1056
  - Tektronix 4014 1056, 1264
  - Xerox 1700 1265
- terminals, multiple virtual 32
- terminating error-logging daemon 404
- terminating tlogger daemon 1071
- test command 934, 1064-1066
- text file, changing the format of 686
- text processing commands
  - checkcw 275
  - checkeq 395
  - checkmm 663
  - col 179
  - cw 275
  - eqn 395
  - eqncheck 395
  - greek 499
  - mant 666
  - mm 663
  - mmt 666
  - mt 666
  - mvt 666
  - neqn 395
  - nroff 709
  - spell 969
  - tbl 1053
  - troff 710
- tic command 1067
- time a command 1068, 1069
- time command 1068
- TIMEOUT 924
- times command 934
- timex command 1069-1070
- title command 1016
- tlog command 1071, 1073
- tlogger command 1072-1073
- tlogger daemon 1071
- toc commands 1074-1076
- tons conversion routine 302, 303
- tosjis conversion routine (Japanese Language Support) 302, 303
- total command 1008
- touch command 1077-1078
- tplot command 1079-1080
- tput command 1081-1082
- tr command 1083-1085
- trace command 1086-1090
- translate characters 1083
- translate profile
- trap command 934
- trcrpt command 1091-1092
- trcstop command 1093
- trcupdate command 1094-1096
- trdiag command 1097-1098
- troff command 710, 720
  - requests removed by the deroff command 313
  - tbl, preprocessor 1053
- true command 1099
- tsh command 1100-1101
- tsort command 1102-1103
- ttoc command 1075
- ttt game 1104

---

tty command 1105-1106  
turn off process accounting 16  
turn on accounting functions 16  
turnacct command 16  
turning the computer off 946  
turnoff command 1107  
turnon command 1107  
tvi command 1108  
type command 934

## U

uadm command 1133-1136  
ugtable command 1109  
ulimit command 934  
umask command 935, 1110-1111  
umount command 1112-1113  
uname command 1114-1115  
unget command 1116-1117  
uniq command 1118  
units command 1119-1121  
unlink 575  
unlink command 576  
unmount command 1112-1113  
unmounting a file system 1112  
unpack command 730-732  
unset command 935  
untab command 1040  
until command 931  
update a program 1127  
update groups of programs 625  
updatep command 1122-1128  
updating access times of a file 1077  
updating modification times of a file 1077  
updating the superblock 1030  
usage, command summary 18  
user  
    adding 1129  
    adm 15  
    changing 1129  
    deleting 1129  
user IDs and names, displaying 517  
user mask, modifying 1110  
userid

adm 848  
    root 552, 848, 1026, 1027  
users command 1129-1132  
using  
    display station features 332  
uuccheck command 1137-1138  
uucico command 1139-1140  
uucleanup command 1141-1143  
uucp command 1144-1148  
uulog command 1149-1150  
uuname command 1151-1152  
uupick command 1153-1155  
uusched command 1156-1157  
uustat command 1158-1161  
uuto command 1162-1163  
uutry command 1164-1165  
uux command 1166-1171  
uuxqt command 1172-1173

## V

val command 1175-1176  
var command 1008  
varyoff command 1177-1179  
varyon command 1180-1181  
vc command 1182-1185  
vcc command 141  
vedit command 1187-1202  
verify command 1186  
version date cutoff, specifying 482  
vi command 1187-1202  
view command 1187-1202  
virtual terminal  
    assigning default display 332  
    assigning physical display 332  
    changing DMA pinned page 333  
    setting active color palette 332  
    setting background colors 332  
    setting fonts 332  
    setting foreground colors 332  
virtual terminal manager 32  
vmh command 1203-1204  
vrmconfig command 1206-1207  
vrmfmt command 141

---

vrm2rtfont command 1205  
vtoc command 1075

## W

wait command 935  
wall command 1208  
watch command 1209  
wc command 1211-1212  
what command 1213-1214  
whatis command 511  
    utility commands 511  
whatnow command 1215-1218  
while command 931  
who command 1219-1221  
whom command 1222-1224  
work station characteristics 1062  
work station parameters, setting 1018  
work stations  
    DASI 300 1262  
    DASI 300s 1262  
    DASI 450 1265  
    Diablo 1620 1265  
    HP2621 514  
    HP2640 514  
    phototypesetter simulator 1056  
    Tektronix 4014 1056, 1264  
    Xerox 1700 1265  
write command 1225-1229  
write operations 22  
writesrv command 1230  
writing audit records 80  
writing buffered files to fixed disk 1030  
writing the last part of a file 1044  
writing to standard output  
    acctcon1 24  
    acctmerg 28  
    acctwtmp 458  
    awk 81

bdiff 102  
cal 132  
cb 139  
cflow 154  
cmp 177  
comb 181  
fwtmp 457  
    wtmpfix 458  
wtmpfix command 458  
wump game 1231

## X

X windows 1236  
xargs command 1232-1235  
xdbx command 1236

## Y

yacc command 1237-1238  
ypbind daemon 1239  
ypinit command 1243  
yppasswd command 1247  
yppasswdd command 1249  
yppoll command 1251  
yppush command 1252  
ypserv daemon 1256  
ypwhich command 1258  
ypxfr command 1260

## Numerics

300 command 1262-1263  
4014 command 1264  
450 command 1265-1266

**Book Title**

**Order No.**

**Book Evaluation Form**

Your comments can help us produce better books. You may use this form to communicate your comments about this book, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you. Please take a few minutes to evaluate this book as soon as you become familiar with it. Circle Y (Yes) or N (No) for each question that applies and give us any information that may improve this book.

Y N Is the purpose of this book clear?  
\_\_\_\_\_  
\_\_\_\_\_

Y N Are the abbreviations and acronyms understandable?  
\_\_\_\_\_  
\_\_\_\_\_

Y N Is the table of contents helpful?  
\_\_\_\_\_  
\_\_\_\_\_

Y N Are the examples clear?  
\_\_\_\_\_  
\_\_\_\_\_

Y N Is the index complete?  
\_\_\_\_\_  
\_\_\_\_\_

Y N Are examples provided where they are needed?  
\_\_\_\_\_  
\_\_\_\_\_

Y N Are the chapter titles and other headings meaningful?  
\_\_\_\_\_  
\_\_\_\_\_

Y N Are the illustrations clear?  
\_\_\_\_\_  
\_\_\_\_\_

Y N Is the information organized appropriately?  
\_\_\_\_\_  
\_\_\_\_\_

Y N Is the format of the book (shape, size, color) effective?  
\_\_\_\_\_  
\_\_\_\_\_

Y N Is the information accurate?  
\_\_\_\_\_  
\_\_\_\_\_

**Other Comments**

What could we do to make this book or the entire set of books for this system easier to use?

Y N Is the information complete?  
\_\_\_\_\_  
\_\_\_\_\_

\_\_\_\_\_  
\_\_\_\_\_

Y N Is only necessary information included?  
\_\_\_\_\_  
\_\_\_\_\_

\_\_\_\_\_  
\_\_\_\_\_

Y N Does the book refer you to the appropriate places for more information?  
\_\_\_\_\_  
\_\_\_\_\_

\_\_\_\_\_  
\_\_\_\_\_

**Optional Information**

Y N Are terms defined clearly?  
\_\_\_\_\_  
\_\_\_\_\_

Your name \_\_\_\_\_

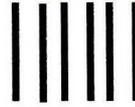
Y N Are terms used consistently?  
\_\_\_\_\_  
\_\_\_\_\_

Company name \_\_\_\_\_

Street address \_\_\_\_\_

City, State, ZIP \_\_\_\_\_

No postage necessary if mailed in the U.S.A.

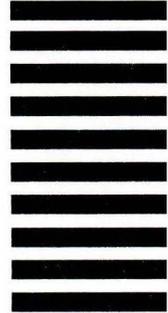


NO POSTAGE  
NECESSARY  
IF MAILED  
IN THE  
UNITED STATES

**BUSINESS REPLY MAIL**  
FIRST CLASS PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation  
Department 997, Building 998  
11400 Burnet Rd.  
Austin, Texas 78758-3493



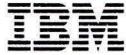
Fold and Tape

nd Tape

Cut or Fold Along Line

Tape

Please Do Not Staple



The IBM AIX/RT Family

**Reader's Comment Form**

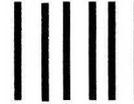
**AIX Operating System  
Commands Reference**

SC23-2011-1/SC23-2081-1

Your comments assist us in improving our products. IBM may use and distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

For prompt resolution to questions regarding set up, operation, program support, and new program literature, contact your IBM representative, your IBM authorized dealer, or your IBM authorized remarketer.

Comments:

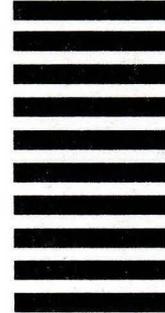


NO POSTAGE  
NECESSARY  
IF MAILED  
IN THE  
UNITED STATES

**BUSINESS REPLY MAIL**  
FIRST CLASS PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation  
Department 997, Building 998  
11400 Burnet Rd.  
Austin, Texas 78758-3493



Cut or Fold Along Line

Fold and Tape

of Tape

Tape

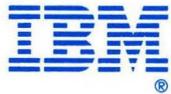
Please Do Not Staple

© IBM Corp. 1988  
All rights reserved.

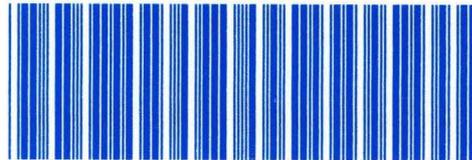
International Business  
Machines Corporation  
11400 Burnet Road  
Austin, Texas 78758

Printed in the  
United States of America

SC23-2081-1



SC23-2081-1



27F4354