```
RRRRRRRRRRR      UUU          UUU  NNN         NNN     000000000     FFFFFFFFFFFFFFF   FFFFFFFFFFFFFFF
RRRRRRRRRRRR     UUU          UUU  NNN         NNN     000000000     FFFFFFFFFFFFFFF   FFFFFFFFFFFFFFF
RRRRRRRRRRRR     UUU          UUU  NNN         NNN     000000000     FFFFFFFFFFFFFFF   FFFFFFFFFFFFFFF
RRR          RRR UUU          UUU  NNN         NNN  000          000 FFF               FFF
RRR          RRR UUU          UUU  NNN         NNN  000          000 FFF               FFF
RRR          RRR UUU          UUU  NNN         NNN  000          000 FFF               FFF
RRR          RRR UUU          UUU  NNNNNN      NNN  000          000 FFF               FFF
RRR          RRR UUU          UUU  NNNNNN      NNN  000          000 FFF               FFF
RRR          RRR UUU          UUU  NNNNNN      NNN  000          000 FFF               FFF
RRRRRRRRRRR      UUU          UUU  NNN   NNN   NNN  000          000 FFFFFFFFFFF       FFFFFFFFFFF
RRRRRRRRRRRR     UUU          UUU  NNN   NNN   NNN  000          000 FFFFFFFFFFF       FFFFFFFFFFF
RRRRRRRRRRRR     UUU          UUU  NNN   NNN   NNN  000          000 FFFFFFFFFFF       FFFFFFFFFFF
RRR   RRR        UUU          UUU  NNN       NNNNNN 000          000 FFF               FFF
RRR   RRR        UUU          UUU  NNN       NNNNNN 000          000 FFF               FFF
RRR   RRR        UUU          UUU  NNN       NNNNNN 000          000 FFF               FFF
RRR      RRR     UUU          UUU  NNN          NNN 000          000 FFF               FFF
RRR      RRR     UUU          UUU  NNN          NNN 000          000 FFF               FFF
RRR      RRR     UUU          UUU  NNN          NNN 000          000 FFF               FFF
RRR         RRR  UUUUUUUUUUUUUUU  NNN          NNN    000000000     FFF               FFF
RRR         RRR  UUUUUUUUUUUUUUU  NNN          NNN    000000000     FFF               FFF
RRR         RRR  UUUUUUUUUUUUUUU  NNN          NNN    000000000     FFF               FFF
RRR         RRR  UUUUUUUUUUUUUUU  NNN          NNN    000000000     FFF               FFF
RRR         RRR  UUUUUUUUUUUUUUU  NNN          NNN    000000000     FFF               FFF
```

```
_$2

Sym
---
NDX
NDX
NUM
NUM
OPE
OUT
PAC
PAC
PAC
PAC
PAC
PAC
PAC
PAD
PAG
PAG
PAG
PAG
PAG
PAG
PAG
PER
PUT
RCO
RIN

RLI
RNO
RNO
RTY
SAV
STR
STR
STR
STR
STR
STR

STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
```

```
PPPPPPPP    EEEEEEEEEE  RRRRRRRR    MM      MM  UU      UU  TTTTTTTTTT  EEEEEEEEEE
PPPPPPPP    EEEEEEEEEE  RRRRRRRR    MM      MM  UU      UU  TTTTTTTTTT  EEEEEEEEEE
PP      PP  EE          RR      RR  MMMM  MMMM  UU      UU      TT      EE
PP      PP  EE          RR      RR  MMMM  MMMM  UU      UU      TT      EE
PP      PP  EE          RR      RR  MM  MM  MM  UU      UU      TT      EE
PP      PP  EE          RR      RR  MM  MM  MM  UU      UU      TT      EE
PPPPPPPP    EEEEEEEE    RRRRRRRR    MM      MM  UU      UU      TT      EEEEEEEE
PPPPPPPP    EEEEEEEE    RRRRRRRR    MM      MM  UU      UU      TT      EEEEEEEE
PP          EE          RR  RR      MM      MM  UU      UU      TT      EE
PP          EE          RR  RR      MM      MM  UU      UU      TT      EE
PP          EE          RR    RR    MM      MM  UU      UU      TT      EE              ....
PP          EE          RR    RR    MM      MM  UU      UU      TT      EE              ....
PP          EEEEEEEEEE  RR      RR  MM      MM  UUUUUUUUUU      TT      EEEEEEEEEE      ....
PP          EEEEEEEEEE  RR      RR  MM      MM  UUUUUUUUUU      TT      EEEEEEEEEE      ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
;     1          0001  0 %TITLE 'PERMUTE -- Permute index entries'
;     2          0002  0 MODULE PERMUTE (IDENT = 'V04-000'
;     3          0003  0                 %BLISS32 [, ADDRESSING_MODE (EXTERNAL = LONG_RELATIVE, NONEXTERNAL = LONG_RELATIVE)]
;     4          0004  0                 ) =
;     5          0005  1 BEGIN
;     6          0006  1 !
;     7          0007  1 !**********************************************************************
;     8          0008  1 !*                                                                    *
;     9          0009  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                           *
;    10          0010  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.            *
;    11          0011  1 !*  ALL RIGHTS RESERVED.                                             *
;    12          0012  1 !*                                                                    *
;    13          0013  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
;    14          0014  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    15          0015  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
;    16          0016  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
;    17          0017  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
;    18          0018  1 !*  TRANSFERRED.                                                       *
;    19          0019  1 !*                                                                    *
;    20          0020  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
;    21          0021  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
;    22          0022  1 !*  CORPORATION.                                                       *
;    23          0023  1 !*                                                                    *
;    24          0024  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
;    25          0025  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.            *
;    26          0026  1 !*                                                                    *
;    27          0027  1 !*                                                                    *
;    28          0028  1 !**********************************************************************
;    29          0029  1 !
;    30          0030  1
;    31          0031  1 !++
;    32          0032  1 ! FACILITY:
;    33          0033  1 !   DSR (Digital Stardard RUNOFF) /DSRPLUS DSRINDEX/INDEX Utility
;    34          0034  1 !
;    35          0035  1 ! ABSTRACT:
;    36          0036  1 !   Routines to permute index entries
;    37          0037  1 !
;    38          0038  1 ! ENVIRONMENT:   Transportable
;    39          0039  1 !
;    40          0040  1 ! AUTHOR:        JPK
;    41          0041  1 !
;    42          0042  1 ! CREATION DATE: December 1981
;    43          0043  1 !
;    44          0044  1 ! MODIFIED BY:
;    45          0045  1 !
;    46          0046  1 !       004     JPK00024        23-May-1983
;    47          0047  1 !               Modified lowercasing algorithm in PERMUTE. Now lowercase only
;    48          0048  1 !               if word contains only 1 letter or if second letter in word is
;    49          0049  1 !               lowercase. Picked up modules PAGMRG and POOL from DSR/DSRPLUS
;    50          0050  1 !               since they are no longer used by DSR/DSRPLUS.
;    51          0051  1 !
;    52          0052  1 !       003     JPK00015        04-Feb-1983
;    53          0053  1 !               Cleaned up module names, modified revision history to
;    54          0054  1 !               conform with established standards. Updated copyright dates.
;    55          0055  1 !
;    56          0056  1 !       002     JPK00014        02-Feb-1983
;    57          0057  1 !               Modified PERMUTE to remove extra whitespace in index entries.
```

```
;    58          0058  1 !                Changed module name.
;    59          0059  1 !
;    60          0060  1 !--
;    61          0061  1
;    62          0062  1 !
;    63          0063  1 ! TABLE OF CONTENTS:
;    64          0064  1 !
;    65          0065  1
;    66          0066  1 FORWARD ROUTINE
;    67          0067  1     PERMUT     : NOVALUE,                  ! Permute index entry
;    68          0068  1     MAKE_ENTRY : NOVALUE,                  ! Make a permuted index entry
;    69          0069  1     FIX_CASE   : NOVALUE,                  ! Fix up case of permuted entry
;    70          0070  1     COPY_STR   : NOVALUE;                  ! Copy string from a to b and filter
;    71          0071  1                                            ! out the nopermute flag.
;    72          0072  1
;    73          0073  1 !
;    74          0074  1 ! INCLUDE FILES:
;    75          0075  1 !
;    76          0076  1
;    77          0077  1 LIBRARY 'NXPORT:XPORT';
;    78          0078  1
;    79          0079  1 SWITCHES LIST (REQUIRE);
;    80          0080  1
;    81          0081  1 REQUIRE 'REQ:NDXXPL';                      ! Extended indexing attributes
```

```
;  R0082  1      !
;  R0083  1      ! Version:      'V04-000'
;  R0084  1      !
;  R0085  1      !****************************************************************************
;  R0086  1      !*                                                                          *
;  R0087  1      !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
;  R0088  1      !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
;  R0089  1      !*  ALL RIGHTS RESERVED.                                                    *
;  R0090  1      !*                                                                          *
;  R0091  1      !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;  R0092  1      !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;  R0093  1      !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;  R0094  1      !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;  R0095  1      !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;  R0096  1      !*  TRANSFERRED.                                                            *
;  R0097  1      !*                                                                          *
;  R0098  1      !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;  R0099  1      !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;  R0100  1      !*  CORPORATION.                                                            *
;  R0101  1      !*                                                                          *
;  R0102  1      !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;  R0103  1      !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
;  R0104  1      !*                                                                          *
;  R0105  1      !*                                                                          *
;  R0106  1      !****************************************************************************
;  R0107  1      !
;  R0108  1
;  R0109  1      !++
;  R0110  1      ! FACILITY:      DSR (Digital Standard RUNOFF) / DSRPLUS
;  R0111  1      !
;  R0112  1      ! ABSTRACT:
;  R0113  1      !       This file contains definitions of data structures used to support
;  R0114  1      !       the extended indexing features of the DSRPLUS INDEX program.
;  R0115  1      !
;  R0116  1      ! ENVIRONMENT:   Transportable BLISS
;  R0117  1      !
;  R0118  1      ! AUTHOR:        J.P. Kellerman
;  R0119  1      !
;  R0120  1      ! CREATION DATE: January 1982
;  R0121  1      !
;  R0122  1      ! MODIFIED BY:
;  R0123  1      !
;  R0124  1      !       002     KAD00002        Keith Dawson    07-Mar-1983
;  R0125  1      !               Global edit of all modules. Updated module names, idents,
;  R0126  1      !               copyright dates. Changed require files to BLISS library.
;  R0127  1      !
;  R0128  1      !--
;  R0129  1
;  R0130  1      ! Extended INDEX attributes block.
;  R0131  1
;  R0132  1      $FIELD XPL_FIELDS =
;  R0133  1          SET
;  R0134  1
;  R0135  1          XPL$V_OPTIONS       = [$INTEGER],           ! Attributes options
;  R0136  1
;  R0137  1              $OVERLAY (XPL$V_OPTIONS)
;  R0138  1
```

```
;  R0139  1              XPL$V_VALID             = [$BIT],       ! Attributes block contains valid information.
;  R0140  1              XPL$V_BOLD              = [$BIT],       ! Bold page reference.
;  R0141  1              XPL$V_UNDERLINE         = [$BIT],       ! Underlined page reference.
;  R0142  1              XPL$V_BEGIN             = [$BIT],       ! Begin page range.
;  R0143  1              XPL$V_END               = [$BIT],       ! End page range.
;  R0144  1              XPL$V_MASTER            = [$BIT],       ! Master index entry.
;  R0145  1              XPL$V_PERMUTE           = [$BIT],       ! Permute index entry.
;  R0146  1              XPL$V_NOPERMUTE         = [$BIT],       ! Set if permute explicitly forbidden.
;  R0147  1              XPL$V_SORT              = [$BIT],       ! Set if SORT string present.
;  R0148  1              XPL$V_APPEND            = [$BIT],       ! Set if append string present.
;  R0149  1
;  R0150  1              $CONTINUE
;  R0151  1
;  R0152  1          XPL$T_SORT          = [$DESCRIPTOR(DYNAMIC)], ! SORT string.
;  R0153  1          XPL$T_APPEND        = [$DESCRIPTOR(DYNAMIC)]  ! APPEND string.
;  R0154  1
;  R0155  1          TES;
;  R0156  1
;  R0157  1      LITERAL
;  R0158  1          XPL$K_LENGTH = $FIELD_SET_SIZE;
;  R0159  1
;  R0160  1      MACRO
;  R0161  1          $XPL_BLOCK = BLOCK [XPL$K_LENGTH] FIELD (XPL_FIELDS) %;
;  R0162  1
;  R0163  1      !
;  R0164  1      ! Macros for INDEX_ATTRIBUTES flags
;  R0165  1      !
;  R0166  1      MACRO
;  R0167  1          XPLUS$V_VALID       = 0, 0, 1, 0 %,  ! Set if attributes data is valid.
;  R0168  1          XPLUS$V_BOLD        = 0, 1, 1, 0 %,  ! Set if page reference is bolded.
;  R0169  1          XPLUS$V_UNDERLINE   = 0, 2, 1, 0 %,  ! Set if page reference is underlined.
;  R0170  1          XPLUS$V_BEGIN       = 0, 3, 1, 0 %,  ! Set if entry begins a page range.
;  R0171  1          XPLUS$V_END         = 0, 4, 1, 0 %,  ! Set if entry ends a page range.
;  R0172  1          XPLUS$V_MASTER      = 0, 5, 1, 0 %,  ! Set if master index entry only.
;  R0173  1          XPLUS$V_PERMUTE     = 0, 6, 1, 0 %,  ! Set if entry is to be permuted.
;  R0174  1          XPLUS$V_NOPERMUTE   = 0, 7, 1, 0 %,  ! Set if permute is explicitly forbidden.
;  R0175  1          XPLUS$V_SORT        = 0, 8, 1, 0 %,  ! Set if entry contains a SORT string.
;  R0176  1          XPLUS$V_APPEND      = 0, 9, 1, 0 %;  ! Set if entry contains an APPEND string.
;  R0177  1
;  R0178  1      !                          End of NDXXPL.REQ
```

```
;   82          0179  1
;   83          0180  1 REQUIRE 'REQ:LETTER';
```

```
;  R0181  1        !
;  R0182  1        ! Version:      'V04-000'
;  R0183  1        !
;  R0184  1        !****************************************************************************
;  R0185  1        !*                                                                          *
;  R0186  1        !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
;  R0187  1        !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
;  R0188  1        !*  ALL RIGHTS RESERVED.                                                    *
;  R0189  1        !*                                                                          *
;  R0190  1        !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;  R0191  1        !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;  R0192  1        !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;  R0193  1        !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;  R0194  1        !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;  R0195  1        !*  TRANSFERRED.                                                            *
;  R0196  1        !*                                                                          *
;  R0197  1        !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;  R0198  1        !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;  R0199  1        !*  CORPORATION.                                                            *
;  R0200  1        !*                                                                          *
;  R0201  1        !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;  R0202  1        !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
;  R0203  1        !*                                                                          *
;  R0204  1        !*                                                                          *
;  R0205  1        !****************************************************************************
;  R0206  1        !
;  R0207  1
;  R0208  1        !++
;  R0209  1        ! FACILITY:      DSR (Digital Standard RUNOFF) / DSRPLUS
;  R0210  1        !
;  R0211  1        ! ABSTRACT:
;  R0212  1        !       Macros to test if a character is an appropriately flavored letter,
;  R0213  1        !       and macros to convert between upper and lower case.
;  R0214  1        !
;  R0215  1        ! ENVIRONMENT:   Transportable BLISS
;  R0216  1        !
;  R0217  1        ! AUTHOR:        Rich Friday
;  R0218  1        !
;  R0219  1        ! CREATION DATE: 1978
;  R0220  1        !
;  R0221  1        ! MODIFIED BY:
;  R0222  1        !
;  R0223  1        !       002     KAD00002        Keith Dawson    07-Mar-1983
;  R0224  1        !               Global edit of all modules. Updated module names, idents,
;  R0225  1        !               copyright dates. Changed require files to BLISS library.
;  R0226  1        !
;  R0227  1        !--
;  R0228  1
;  R0229  1        MACRO
; MR0230  1            upper_letter (khar) =      ! See if upper case letter
; MR0231  1                (khar GEQ %C'A' AND khar LEQ %C'Z')
;  R0232  1            %,
;  R0233  1
; MR0234  1            lower_letter (khar) =      ! See if lower case letter
; MR0235  1                (khar GEQ %C'a' AND khar LEQ %C'z')
;  R0236  1            %,
;  R0237  1
```

```
;  MR0238  1          letter (khar)                =       ! See if any type of letter
;  MR0239  1              (upper_letter (khar) OR lower_letter (khar))
;   R0240  1          %;
;   R0241  1
;   R0242  1      MACRO
;  MR0243  1          upper_case (khar)    =       ! Convert to upper case
;  MR0244  1              (khar + %C'A' - %C'a')
;   R0245  1          %,
;   R0246  1
;  MR0247  1          lower_case (khar)    =       ! Convert to lower case
;  MR0248  1              (khar + %C'a' - %C'A')
;   R0249  1          %;
;   R0250  1
;   R0251  1      !                       End of LETTER.REQ
```

```
;    84          0252  1
;    85          0253  1 SWITCHES LIST (NOREQUIRE);
;    86          0254  1
;    87          0255  1 !
;    88          0256  1 ! MACROS:
;    89          0257  1 !
;    90          0258  1 !
;    91          0259  1 ! EQUATED SYMBOLS:
;    92          0260  1 !
;    93          0261  1
;    94          0262  1 LITERAL
;    95          0263  1     TRUE = 1,
;    96          0264  1     FALSE = 0;
;    97          0265  1
;    98          0266  1 !
;    99          0267  1 ! OWN STORAGE:
;   100          0268  1 !
;   101          0269  1 !
;   102          0270  1 ! EXTERNAL REFERENCES:
;   103          0271  1 !
;   104          0272  1
;   105          0273  1 EXTERNAL LITERAL
;   106          0274  1     TAB    : UNSIGNED (8),          ! TAB character
;   107          0275  1     RINTES : UNSIGNED (8);          ! RUNOFF escape sequence
;   108          0276  1
;   109          0277  1 EXTERNAL
;   110          0278  1     XPLBLK : $XPL_BLOCK;            ! Extended indexing attributes block
;   111          0279  1
;   112          0280  1 EXTERNAL ROUTINE
;   113          0281  1     XOUT : NOVALUE;                 ! Insert index entry into internal pool
;   114          0282  1
```

```
;  116     0283  1 %SBTTL 'PERMUT -- Permute Index Entry'
;  117     0284  1 GLOBAL ROUTINE PERMUT (ENTRY_LEN, ENTRY_PTR, XTN, BAR_FLAG) : NOVALUE =
;  118     0285  1 !++
;  119     0286  1 !
;  120     0287  1 ! FUNCTIONAL DESCRIPTION:
;  121     0288  1 !
;  122     0289  1 !       This routine is called to permute and output index entries
;  123     0290  1 !
;  124     0291  1 ! FORMAL PARAMETERS:
;  125     0292  1 !
;  126     0293  1 !       ENTRY_LEN - Number of characters in entry
;  127     0294  1 !       ENTRY_PTR - A CH$PTR to index entry
;  128     0295  1 !       XTN       - Transaction number
;  129     0296  1 !       BAR_FLAG  - Change bar flag
;  130     0297  1 !
;  131     0298  1 ! IMPLICIT INPUTS:
;  132     0299  1 !
;  133     0300  1 !       XPLBLK    - Extended indexing attributes block
;  134     0301  1 !
;  135     0302  1 ! IMPLICIT OUTPUTS:
;  136     0303  1 !
;  137     0304  1 !       None
;  138     0305  1 !
;  139     0306  1 ! ROUTINE VALUE:
;  140     0307  1 ! COMPLETION CODES:
;  141     0308  1 !
;  142     0309  1 !       None
;  143     0310  1 !
;  144     0311  1 ! SIDE EFFECTS:
;  145     0312  1 !
;  146     0313  1 !       None
;  147     0314  1 !--
;  148     0315  1
;  149     0316  2     BEGIN
;  150     0317  2
;  151     0318  2     LOCAL
;  152     0319  2         WHITE_SPACE,                                ! Length of string up to white space
;  153     0320  2         SOURCE_PTR,                                 ! Pointer to source string
;  154     0321  2         PERMUTE_PTR,                                ! Pointer to permuted text
;  155     0322  2         PERMUTE_LEN,                                ! Length of permuted text
;  156     0323  2         PERMUTE_STR : VECTOR [CH$ALLOCATION (1200)];    ! Put permuted text here
;  157     0324  2
;  158     0325  2     IF .ENTRY_LEN EQL 0 THEN RETURN;                ! Nothing to do if no text
;  159     0326  2
;  160     0327  2     !
;  161     0328  2     ! Initialization
;  162     0329  2     !
;  163     0330  2     WHITE_SPACE = -1;
;  164     0331  2     SOURCE_PTR = .ENTRY_PTR;                        ! Copy pointer to string
;  165     0332  2     PERMUTE_PTR = CH$PTR (PERMUTE_STR);
;  166     0333  2     PERMUTE_LEN = 0;
;  167     0334  2
;  168     0335  2     IF .XPLBLK [XPL$V_VALID] AND .XPLBLK [XPL$V_PERMUTE]
;  169     0336  2     THEN
;  170     0337  3         BEGIN
;  171     0338  3         !
;  172     0339  3         ! Permute index entry
```

```
;  173     0340  3          !
;  174     0341  3          LOCAL
;  175     0342  3              PERMUTE_FIRST,                  ! TRUE if can permute on first word
;  176     0343  3              FIRST_CHAR_SEEN;                ! TRUE after 1st character found
;  177     0344  3
;  178     0345  3          PERMUTE_FIRST = TRUE;
;  179     0346  3          FIRST_CHAR_SEEN = FALSE;
;  180     0347  3
;  181     0348  3          INCR I FROM 1 TO .ENTRY_LEN DO
;  182     0349  4              BEGIN
;  183     0350  4
;  184     0351  4              LOCAL
;  185     0352  4                  CH;                                 ! Current character
;  186     0353  4
;  187     0354  4              CH = CH$RCHAR_A (SOURCE_PTR);
;  188     0355  4
;  189     0356  4              SELECTONE .CH OF
;  190     0357  4                  SET
;  191     0358  4
;  192     0359  4                  [RINTES]:
;  193     0360  5                      BEGIN
;  194     0361  5                      !
;  195     0362  5                      ! Process RUNOFF escape sequence
;  196     0363  5                      !
;  197     0364  5                      LOCAL
;  198     0365  5                          FUNCTION_CODE,
;  199     0366  5                          OPERAND;
;  200     0367  5
;  201     0368  5                      FUNCTION_CODE = CH$RCHAR_A (SOURCE_PTR);
;  202     0369  5                      OPERAND = CH$RCHAR_A (SOURCE_PTR);
;  203     0370  5                      I = .I + 2;
;  204     0371  5
;  205     0372  5                      SELECTONE .FUNCTION_CODE OF
;  206     0373  5                          SET
;  207     0374  5
;  208     0375  5                          [%C'J'] :
;  209     0376  6                              BEGIN
;  210     0377  6                              !
;  211     0378  6                              ! A word mark. For index entries this starts a
;  212     0379  6                              ! subindex entry. Since subindex entries are not
;  213     0380  6                              ! permuted, copy subindex entries and exit.
;  214     0381  6                              !
;  215     0382  6                              COPY_STR (3 + .ENTRY_LEN - .I,
;  216     0383  6                                  CH$PLUS (.SOURCE_PTR, -3),
;  217     0384  6                                  PERMUTE_LEN, PERMUTE_PTR);
;  218     0385  6
;  219     0386  6                              EXITLOOP;
;  220     0387  5                              END;
;  221     0388  5
;  222     0389  5                          [%C'P']:
;  223     0390  5                              !
;  224     0391  5                              ! Nopermute flag.
;  225     0392  5                              !
;  226     0393  5                              ! If we haven't seen the first character yet, set
;  227     0394  5                              ! flag saying that we can't permute on the first word.
;  228     0395  5                              !
;  229     0396  5                              ! Remove escape sequence by doing nothing.
```

```
;  230     0397  5                        !
;  231     0398  5                        IF NOT .FIRST_CHAR_SEEN THEN PERMUTE_FIRST = FALSE;
;  232     0399  5
;  233     0400  5                    [OTHERWISE] :
;  234     0401  6                        BEGIN
;  235     0402  6                        !
;  236     0403  6                        ! Other RUNOFF escape sequences signal a new word
;  237     0404  6                        ! if white space was seen. In this case, It's time
;  238     0405  6                        ! to make a permuted entry. Once the entry has been
;  239     0406  6                        ! made (or if it's not necessary), the escape
;  240     0407  6                        ! sequence is copied to the permuted text string.
;  241     0408  6                        !
;  242     0409  6
;  243     0410  6                        IF .WHITE_SPACE GEQ 0
;  244     0411  6                        THEN
;  245     0412  6                            MAKE_ENTRY (3 + .ENTRY_LEN - .I,
;  246     0413  6                                CH$PLUS (.SOURCE_PTR, -3),
;  247     0414  6                                .WHITE_SPACE, CH$PTR (PERMUTE_STR),
;  248     0415  6                                .XTN, .BAR_FLAG);
;  249     0416  6
;  250     0417  6                        CH$WCHAR_A (.CH, PERMUTE_PTR);
;  251     0418  6                        CH$WCHAR_A (.FUNCTION_CODE, PERMUTE_PTR);
;  252     0419  6                        CH$WCHAR_A (.OPERAND, PERMUTE_PTR);
;  253     0420  6                        PERMUTE_LEN = .PERMUTE_LEN + 3;
;  254     0421  5                        END;
;  255     0422  5
;  256     0423  5                    TES;
;  257     0424  5
;  258     0425  5                WHITE_SPACE = -1;           ! Blank or tab not seen
;  259     0426  4                END;
;  260     0427  4
;  261     0428  4            [%C' ', TAB]:
;  262     0429  5                BEGIN
;  263     0430  5                !
;  264     0431  5                ! Blank or TAB.
;  265     0432  5                ! Remove extra whitespace by inserting a blank for
;  266     0433  5                ! only the first one seen.
;  267     0434  5                !
;  268     0435  5                IF .WHITE_SPACE EQL -1
;  269     0436  5                THEN
;  270     0437  6                    BEGIN
;  271     0438  6                    !
;  272     0439  6                    ! No blanks or tabs seen yet. Save position and insert
;  273     0440  6                    ! a blank in the output string.
;  274     0441  6                    !
;  275     0442  6                    WHITE_SPACE = .PERMUTE_LEN;
;  276     0443  6                    CH$WCHAR_A (%C' ', PERMUTE_PTR);
;  277     0444  6                    PERMUTE_LEN = .PERMUTE_LEN + 1;
;  278     0445  5                    END;
;  279     0446  5
;  280     0447  4                END;
;  281     0448  4
;  282     0449  4            [OTHERWISE]:
;  283     0450  5                BEGIN
;  284     0451  5                !
;  285     0452  5                ! Process normal character
;  286     0453  5                !
```

```
;  287     0454  5                    FIRST_CHAR_SEEN = TRUE;
;  288     0455  5                    CH$WCHAR_A (.CH, PERMUTE_PTR);
;  289     0456  5                    PERMUTE_LEN = .PERMUTE_LEN + 1;
;  290     0457  5
;  291     0458  5                    IF .WHITE_SPACE GEQ 0
;  292     0459  5                    THEN
;  293     0460  5                        !
;  294     0461  5                        ! White space was seen before this character. Hence we
;  295     0462  5                        ! are starting a new word and it is time to permute.
;  296     0463  5                        !
;  297     0464  5                        MAKE_ENTRY (.ENTRY_LEN - .I + 1,
;  298     0465  5                            CH$PLUS (.SOURCE_PTR, -1), .WHITE_SPACE,
;  299     0466  5                            CH$PTR (PERMUTE_STR), .XTN, .BAR_FLAG);
;  300     0467  5
;  301     0468  5                    WHITE_SPACE = -1;           ! Blank or tab not seen
;  302     0469  4                    END;
;  303     0470  4
;  304     0471  4                TES;
;  305     0472  4
;  306     0473  3            END;
;  307     0474  3
;  308     0475  3        IF .PERMUTE_FIRST
;  309     0476  3        THEN
;  310     0477  4            BEGIN
;  311     0478  4            FIX_CASE (.PERMUTE_LEN, CH$PTR (PERMUTE_STR), 0, 0);
;  312     0479  4            XOUT (.PERMUTE_LEN, CH$PTR (PERMUTE_STR), .XTN, .BAR_FLAG);
;  313     0480  3            END;
;  314     0481  3        END
;  315     0482  2    ELSE
;  316     0483  3        BEGIN
;  317     0484  3        !
;  318     0485  3        ! Do not permute index entry.
;  319     0486  3        ! Copy it filtering out nopermute flags, insert it in the internal
;  320     0487  3        ! pool and return.
;  321     0488  3        !
;  322     0489  3        COPY_STR (.ENTRY_LEN, .ENTRY_PTR, PERMUTE_LEN, PERMUTE_PTR);
;  323     0490  3        XOUT (.PERMUTE_LEN, CH$PTR (PERMUTE_STR), .XTN, .BAR_FLAG);
;  324     0491  2        END;
;  325     0492  2
;  326     0493  1    END;


                                                    .TITLE  PERMUTE PERMUTE -- Permute index entries
                                                    .IDENT  \V04-000\

                                                    .EXTRN  TAB, RINTES, XPLBLK
                                                    .EXTRN  XOUT

                                                    .PSECT  $CODE$,NOWRT,2

                                    0FFC 00000      .ENTRY  PERMUT, Save R2,R3,R4,R5,R6,R7,R8,R9,R10,-  ; 0284
                                                            R11                                         ;
                5B 00000000V    EF  9E 00002        MOVAB   COPY_STR, R11                               ;
                5E      FB48    CE  9E 00009        MOVAB   -1208(SP), SP                               ;
                55        04    AC  D0 0000E        MOVL    ENTRY_LEN, R5                               ; 0325
                                01  12 00012        BNEQ    1$                                          ;
                                    04 00014        RET                                                 ;
```

```
                        54           01   CE 00015 1$:     MNEGL   #1, WHITE_SPACE                          ; 0330
                        52     08    AC   D0 00018         MOVL    ENTRY_PTR, SOURCE_PTR                    ; 0331
                        6E     08    AE   9E 0001C         MOVAB   PERMUTE_STR, PERMUTE_PTR                 ; 0332
                               04    AE   D4 00020         CLRL    PERMUTE_LEN                              ; 0333
                        03 00000000G EF   E8 00023         BLBS    XPLBLK, 3$                               ; 0335
                                     00F8 31 0002A 2$:     BRW     15$                                      ;
                  F5 00000000G  EF         06 E1 0002D 3$:  BBC     #6, XPLBLK, 2$                           ;
                        59           01   D0 00035         MOVL    #1, PERMUTE_FIRST                        ; 0345
                                     58   D4 00038         CLRL    FIRST_CHAR_SEEN                          ; 0346
                                     56   D4 0003A         CLRL    I                                        ; 0348
                                     00CC 31 0003C         BRW     13$                                      ;
                        53           82   9A 0003F 4$:     MOVZBL  (SOURCE_PTR)+, CH                        ; 0354
                     00000000G  8F         53   D1 00042         CMPL    CH, #RINTES                      ; 0359
                                           6D   12 00049         BNEQ    9$                               ;
                        57           82   9A 0004B         MOVZBL  (SOURCE_PTR)+, FUNCTION_CODE             ; 0368
                        5A           82   9A 0004E         MOVZBL  (SOURCE_PTR)+, OPERAND                   ; 0369
                        56           02   C0 00051         ADDL2   #2, I                                    ; 0370
                     0000004A  8F         57   D1 00054         CMPL    FUNCTION_CODE, #74               ; 0375
                                          15   12 0005B         BNEQ    5$                               ;
                                          5E   DD 0005D         PUSHL   SP                               ; 0383
                                    08    AE   9F 0005F         PUSHAB  PERMUTE_LEN                      ;
                                    FD    A2   9F 00062         PUSHAB  -3(SOURCE_PTR)                   ;
                  50                55         56   C3 00065         SUBL3   I, R5, R0                        ; 0382
                                    03    A0   9F 00069         PUSHAB  3(R0)                            ;
                        6B                 04   FB 0006C         CALLS   #4, COPY_STR                     ; 0383
                                     009F 31   0006F         BRW     14$                              ; 0376
                     00000050  8F         57   D1 00072 5$:     CMPL    FUNCTION_CODE, #80               ; 0389
                                          07   12 00079         BNEQ    6$                               ;
                        38                 58   E8 0007B         BLBS    FIRST_CHAR_SEEN, 8$              ; 0398
                                           59   D4 0007E         CLRL    PERMUTE_FIRST                    ;
                                           34   11 00080         BRB     8$                               ;
                                           54   D5 00082 6$:     TSTL    WHITE_SPACE                      ; 0410
                                           1A   19 00084         BLSS    7$                               ;
                              7E     0C    AC   7D 00086         MOVQ    XTN, -(SP)                       ; 0415
                                    10    AE   9F 0008A         PUSHAB  PERMUTE_STR                      ; 0414
                                          54   DD 0008D         PUSHL   WHITE_SPACE                      ;
                                    FD    A2   9F 0008F         PUSHAB  -3(SOURCE_PTR)                   ; 0413
                  50                55         56   C3 00092         SUBL3   I, R5, R0                        ; 0412
                                    03    A0   9F 00096         PUSHAB  3(R0)                            ;
                     00000000V  EF         06   FB 00099         CALLS   #6, MAKE_ENTRY                   ; 0413
                           00   BE         53   90 000A0 7$:     MOVB    CH, @PERMUTE_PTR                 ; 0417
                                          6E   D6 000A4         INCL    PERMUTE_PTR                      ;
                           00   BE         57   90 000A6         MOVB    FUNCTION_CODE, @PERMUTE_PTR      ; 0418
                                          6E   D6 000AA         INCL    PERMUTE_PTR                      ;
                           00   BE         5A   90 000AC         MOVB    OPERAND, @PERMUTE_PTR            ; 0419
                                          6E   D6 000B0         INCL    PERMUTE_PTR                      ;
                           04   AE         03   C0 000B2         ADDL2   #3, PERMUTE_LEN                  ; 0420
                                          50   11 000B6 8$:     BRB     12$                              ; 0425
                                 20        53   D1 000B8 9$:     CMPL    CH, #32                          ; 0428
                                           09   13 000BB         BEQL    10$                              ;
                     00000000G  8F         53   D1 000BD         CMPL    CH, #TAB                         ;
                                           18   12 000C4         BNEQ    11$                              ;
                     FFFFFFFF   8F         54   D1 000C6 10$:    CMPL    WHITE_SPACE, #-1                 ; 0435
                                           3C   12 000CD         BNEQ    13$                              ;
                                 54    04    AE   D0 000CF         MOVL    PERMUTE_LEN, WHITE_SPACE         ; 0442
                           00   BE         20   90 000D3         MOVB    #32, @PERMUTE_PTR                ; 0443
                                          6E   D6 000D7         INCL    PERMUTE_PTR                      ;
```

```
                               04   AE  D6 000D9         INCL     PERMUTE_LEN                ; 0444
                                    2D  11 000DC         BRB      13$                        ; 0356
                          58        01  D0 000DE 11$:    MOVL     #1, FIRST_CHAR_SEEN        ; 0454
                     00   BE        53  90 000E1         MOVB     CH, @PERMUTE_PTR           ; 0455
                                    6E  D6 000E5         INCL     PERMUTE_PTR                ;
                               04   AE  D6 000E7         INCL     PERMUTE_LEN                ; 0456
                                    54  D5 000EA         TSTL     WHITE_SPACE                ; 0458
                                    1A  19 000EC         BLSS     12$                        ;
                          7E   0C   AC  7D 000EE         MOVQ     XTN, -(SP)                 ; 0466
                               10   AE  9F 000F2         PUSHAB   PERMUTE_STR                ;
                                    54  DD 000F5         PUSHL    WHITE_SPACE                ; 0465
                               FF   A2  9F 000F7         PUSHAB   -1(SOURCE_PTR)             ;
              50          55        56  C3 000FA         SUBL3    I, R5, R0                  ; 0464
                               01   A0  9F 000FE         PUSHAB   1(R0)                      ;
              00000000V   EF        06  FB 00101         CALLS    #6, MAKE_ENTRY             ; 0465
                          54        01  CE 00108 12$:    MNEGL    #1, WHITE_SPACE            ; 0468
       FF2E   56          01        55  F1 0010B 13$:    ACBL     R5, #1, I, 4$              ; 0348
                          2F        59  E9 00111 14$:    BLBC     PERMUTE_FIRST, 17$         ; 0475
                                    7E  7C 00114         CLRQ     -(SP)                      ; 0478
                               10   AE  9F 00116         PUSHAB   PERMUTE_STR                ;
                               10   AE  DD 00119         PUSHL    PERMUTE_LEN                ;
              00000000V   EF        04  FB 0011C         CALLS    #4, FIX_CASE               ;
                                    0D  11 00123         BRB      16$                        ; 0479
                                    5E  DD 00125 15$:    PUSHL    SP                         ; 0489
                               08   AE  9F 00127         PUSHAB   PERMUTE_LEN                ;
                               08   AC  DD 0012A         PUSHL    ENTRY_PTR                  ;
                                    55  DD 0012D         PUSHL    R5                         ;
                          6B        04  FB 0012F         CALLS    #4, COPY_STR               ;
                          7E   0C   AC  7D 00132 16$:    MOVQ     XTN, -(SP)                 ; 0490
                               10   AE  9F 00136         PUSHAB   PERMUTE_STR                ;
                               10   AE  DD 00139         PUSHL    PERMUTE_LEN                ;
              00000000G   EF        04  FB 0013C         CALLS    #4, XOUT                   ;
                                        04 00143 17$:    RET                                 ; 0493

; Routine Size:  324 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  328          0494  1 %SBTTL 'MAKE_ENTRY -- Make a permuted entry'
;  329          0495  1 ROUTINE MAKE_ENTRY (SOURCE_LEN, SOURCE_PTR, PERMUTED_LEN, PERMUTED_PTR, XTN, BAR_FLAG) : NOVALUE =
;  330          0496  1 !++
;  331          0497  1 !
;  332          0498  1 ! FUNCTIONAL DESCRIPTION:
;  333          0499  1 !
;  334          0500  1 !       This routine makes a permuted index entry.
;  335          0501  1 !
;  336          0502  1 !       It copies from the string described by SOURCE_LEN and SOURCE_PTR
;  337          0503  1 !       until either a subindex entry or end of string is encountered.
;  338          0504  1 !       At this point, it appends a comma and a blank and then copies
;  339          0505  1 !       from the permuted string described by PERMUTED_LEN and PERMUTED_PTR.
;  340          0506  1 !       If the break was caused by a subindex entry, the subindex entry
;  341          0507  1 !       is then copied.
;  342          0508  1 !
;  343          0509  1 !       Once the permuted text has been built, it is written out.
;  344          0510  1 !
;  345          0511  1 ! FORMAL PARAMETERS:
;  346          0512  1 !
;  347          0513  1 !       SOURCE_LEN      - Length of unpermuted portion of index entry
;  348          0514  1 !       SOURCE_PTR      - CH$PTR to unpermuted portion of index entry
;  349          0515  1 !       PERMUTED_LEN    - Length of permuted portion of index entry
;  350          0516  1 !       PERMUTED_PTR    - CH$PTR to permuted portion of index entry
;  351          0517  1 !       XTN             - Transaction number
;  352          0518  1 !       BAR_FLAG        - Change bar flag
;  353          0519  1 !
;  354          0520  1 ! IMPLICIT INPUTS:
;  355          0521  1 !
;  356          0522  1 !       None
;  357          0523  1 !
;  358          0524  1 ! IMPLICIT OUTPUTS:
;  359          0525  1 !
;  360          0526  1 !       None
;  361          0527  1 !
;  362          0528  1 ! ROUTINE VALUE:
;  363          0529  1 ! COMPLETION CODES:
;  364          0530  1 !
;  365          0531  1 !       None
;  366          0532  1 !
;  367          0533  1 ! SIDE EFFECTS:
;  368          0534  1 !
;  369          0535  1 !       None
;  370          0536  1 !--
;  371          0537  1
;  372          0538  2     BEGIN
;  373          0539  2
;  374          0540  2     LOCAL
;  375          0541  2         WHITE_SPACE,                                ! To delete trailing white space
;  376          0542  2         PTR,                                        ! Copy of SOURCE_PTR
;  377          0543  2         LINE : VECTOR [CH$ALLOCATION (1200)],       ! Build permuted string here
;  378          0544  2         L_PTR,                                      ! CH$PTR into LINE
;  379          0545  2         L_LEN;                                      ! Length of string in LINE
;  380          0546  2
;  381          0547  2     !
;  382          0548  2     ! Initialization
;  383          0549  2     !
;  384          0550  2     WHITE_SPACE = -1;
```

```
2	PTR = .SOURCE_PTR;
2	L_PTR = CH$PTR (LINE);
2	L_LEN = 0;
2
2	INCR I FROM 1 TO .SOURCE_LEN DO
3	    BEGIN
3
3	    LOCAL
3	        CH;                                     ! Current character
3
3	    CH = CH$RCHAR_A (PTR);
3
3	    SELECTONE .CH OF
3	        SET
3
3	        [RINTES]:
4	            BEGIN
4	            !
4	            ! RUNOFF escape sequence
4	            !
4	            LOCAL
4	                FUNCTION_CODE,
4	                OPERAND;
4
4	            FUNCTION_CODE = CH$RCHAR_A (PTR);
4	            OPERAND = CH$RCHAR_A (PTR);
4	            I = .I + 2;
4
4	            SELECTONE .FUNCTION_CODE OF
4	                SET
4
4	                [%C'J'] :
5	                    BEGIN
5	                    !
5	                    ! Subindex flag. Delete trailing white space,
5	                    ! and append comma, blank
5	                    !
5	                    IF .WHITE_SPACE NEQ -1
5	                    THEN
6	                        BEGIN
6	                        L_PTR = CH$PLUS (.L_PTR, (.WHITE_SPACE - .L_LEN));
6	                        L_LEN = .WHITE_SPACE;
5	                        END;
5
5	                    CH$WCHAR_A (%C',', L_PTR);
5	                    CH$WCHAR_A (%C' ', L_PTR);
5	                    L_LEN = .L_LEN + 2;
5
5	                    !
5	                    ! Raise case of first letter in permuted entry and
5	                    ! drop case of original first letter if necessary.
5	                    !
5	                    FIX_CASE (.L_LEN, CH$PTR (LINE), .PERMUTED_LEN, .PERMUTED_PTR);
5
5	                    !
5	                    ! Finally, append permuted text to end of line,
5	                    ! append the subindex entry and output.
```

```
;  442     0608  5                          !
;  443     0609  5                          COPY_STR (.PERMUTED_LEN, .PERMUTED_PTR, L_LEN, L_PTR);
;  444     0610  5                          COPY_STR (3 + .SOURCE_LEN - .I, CH$PLUS (.PTR, -3), L_LEN, L_PTR);
;  445     0611  5                          XOUT (.L_LEN, CH$PTR (LINE), .XTN, .BAR_FLAG);
;  446     0612  5                          RETURN;
;  447     0613  4                          END;
;  448     0614  4
;  449     0615  4                      [%C'P']:
;  450     0616  4                          !
;  451     0617  4                          ! Filter out nopermute flags by doing nothing
;  452     0618  4                          !
;  453     0619  4                          ;
;  454     0620  4
;  455     0621  4                      [OTHERWISE] :
;  456     0622  5                          BEGIN
;  457     0623  5                          !
;  458     0624  5                          ! Other RUNOFF escape sequences are just copied
;  459     0625  5                          !
;  460     0626  5                          CH$WCHAR_A (.CH, L_PTR);
;  461     0627  5                          CH$WCHAR_A (.FUNCTION_CODE, L_PTR);
;  462     0628  5                          CH$WCHAR_A (.OPERAND, L_PTR);
;  463     0629  5                          L_LEN = .L_LEN + 3;
;  464     0630  4                          END;
;  465     0631  4
;  466     0632  4                      TES;
;  467     0633  4
;  468     0634  4                  WHITE_SPACE = -1;
;  469     0635  3                  END;
;  470     0636  3
;  471     0637  3              [%C' ', TAB]:
;  472     0638  4                  BEGIN
;  473     0639  4                  !
;  474     0640  4                  ! Blank or TAB.
;  475     0641  4                  ! Remove extra whitespace by inserting a blank for
;  476     0642  4                  ! only the first one seen.
;  477     0643  4                  !
;  478     0644  4                  IF .WHITE_SPACE EQL -1
;  479     0645  4                  THEN
;  480     0646  5                      BEGIN
;  481     0647  5                      !
;  482     0648  5                      ! No blanks or tabs seen yet. Save position and insert a
;  483     0649  5                      ! blank in the output string.
;  484     0650  5                      !
;  485     0651  5                      WHITE_SPACE = .L_LEN;
;  486     0652  5                      CH$WCHAR_A (%C' ', L_PTR);
;  487     0653  5                      L_LEN = .L_LEN + 1;
;  488     0654  4                      END;
;  489     0655  4
;  490     0656  3                  END;
;  491     0657  3
;  492     0658  3              [OTHERWISE]:
;  493     0659  4                  BEGIN
;  494     0660  4                  !
;  495     0661  4                  ! A normal character
;  496     0662  4                  !
;  497     0663  4                  WHITE_SPACE = -1;
;  498     0664  4                  CH$WCHAR_A (.CH, L_PTR);
```

```
;  499          0665  4                  L_LEN = .L_LEN + 1;
;  500          0666  3                  END;
;  501          0667  3
;  502          0668  3              TES;
;  503          0669  3
;  504          0670  2          END;
;  505          0671  2
;  506          0672  2          !
;  507          0673  2          ! End of input string reached. Delete trailing white space,
;  508          0674  2          ! append comma, blank
;  509          0675  2          !
;  510          0676  2          IF .WHITE_SPACE NEQ -1
;  511          0677  2          THEN
;  512          0678  3              BEGIN
;  513          0679  3              L_PTR = CH$PLUS (.L_PTR, (.WHITE_SPACE - .L_LEN));
;  514          0680  3              L_LEN = .WHITE_SPACE;
;  515          0681  2              END;
;  516          0682  2
;  517          0683  2          CH$WCHAR_A (%C',', L_PTR);
;  518          0684  2          CH$WCHAR_A (%C' ', L_PTR);
;  519          0685  2          L_LEN = .L_LEN + 2;
;  520          0686  2
;  521          0687  2          !
;  522          0688  2          ! Raise case of first letter in permuted entry and
;  523          0689  2          ! drop case of original first letter if necessary.
;  524          0690  2          !
;  525          0691  2          FIX_CASE (.L_LEN, CH$PTR (LINE), .PERMUTED_LEN, .PERMUTED_PTR);
;  526          0692  2
;  527          0693  2          !
;  528          0694  2          ! Finally, append permuted text to end of line and output.
;  529          0695  2          !
;  530          0696  2          COPY_STR (.PERMUTED_LEN, .PERMUTED_PTR, L_LEN, L_PTR);
;  531          0697  2          XOUT (.L_LEN, CH$PTR (LINE), .XTN, .BAR_FLAG);
;  532          0698  1          END;


                                 00FC 00000 MAKE_ENTRY:
                                                    .WORD   Save R2,R3,R4,R5,R6,R7                  ; 0495
                      57 00000000V   EF   9E 00002          MOVAB   COPY_STR, R7                    ;
                      56 00000000V   EF   9E 00009          MOVAB   FIX_CASE, R6                    ;
                      5E      FB4C   CE   9E 00010          MOVAB   -1204(SP), SP                   ;
                      52             01   CE 00015          MNEGL   #1, WHITE_SPACE                 ; 0550
                      53          08   AC   D0 00018          MOVL    SOURCE_PTR, PTR                 ; 0551
                                  04   AE   9F 0001C          PUSHAB  LINE                            ; 0552
                                  04   AE   D4 0001F          CLRL    L_LEN                           ; 0553
                                       54   D4 00022          CLRL    I                               ; 0555
                                  00C3 31   00024          BRW     10$                             ;
                      50             83   9A 00027 1$:      MOVZBL  (PTR)+, CH                      ; 0561
00000000G             8F             50   D1 0002A          CMPL    CH, #RINTES                     ; 0566
                                     03   13 00031          BEQL    2$                              ;
                                  0087 31   00033          BRW     6$                              ;
                      51             83   9A 00036 2$:      MOVZBL  (PTR)+, FUNCTION_CODE           ; 0575
                      55             83   9A 00039          MOVZBL  (PTR)+, OPERAND                 ; 0576
                      54             02   C0 0003C          ADDL2   #2, I                           ; 0577
```

```
                0000004A   8F              51   D1 0003F          CMPL    FUNCTION_CODE, #74                      ; 0582
                                           51   12 00046          BNEQ    4$
                FFFFFFFF   8F              52   D1 00048          CMPL    WHITE_SPACE, #-1                        ; 0588
                                           0C   13 0004F          BEQL    3$
          50               52        04    AE   C3 00051          SUBL3   L_LEN, WHITE_SPACE, R0                  ; 0591
                           6E              50   C0 00056          ADDL2   R0, L_PTR
                     04    AE              52   D0 00059          MOVL    WHITE_SPACE, L_LEN                      ; 0592
                     00    BE              2C   90 0005D 3$:      MOVB    #44, @L_PTR                             ; 0595
                                           6E   D6 00061          INCL    L_PTR
                     00    BE              20   90 00063          MOVB    #32, @L_PTR                             ; 0596
                                           6E   D6 00067          INCL    L_PTR
                     04    AE              02   C0 00069          ADDL2   #2, L_LEN                               ; 0597
                           7E        0C    AC   7D 0006D          MOVQ    PERMUTED_LEN, -(SP)                     ; 0603
                                     10    AE   9F 00071          PUSHAB  LINE
                                     10    AE   DD 00074          PUSHL   L_LEN
                           66              04   FB 00077          CALLS   #4, FIX_CASE
                                           5E   DD 0007A          PUSHL   SP                                      ; 0609
                                     08    AE   9F 0007C          PUSHAB  L_LEN
                           7E        0C    AC   7D 0007F          MOVQ    PERMUTED_LEN, -(SP)
                           67              04   FB 00083          CALLS   #4, COPY_STR
                                           5E   DD 00086          PUSHL   SP                                      ; 0610
                                     08    AE   9F 00088          PUSHAB  L_LEN
                                     FD    A3   9F 0008B          PUSHAB  -3(PTR)
          50         04    AC              54   C3 0008E          SUBL3   I, SOURCE_LEN, R0
                                     03    A0   9F 00093          PUSHAB  3(R0)
                                         0093   31 00096          BRW     12$
                00000050   8F              51   D1 00099 4$:      CMPL    FUNCTION_CODE, #80                      ; 0615
                                           16   13 000A0          BEQL    5$
                     00    BE              50   90 000A2          MOVB    CH, @L_PTR                              ; 0626
                                           6E   D6 000A6          INCL    L_PTR
                     00    BE              51   90 000A8          MOVB    FUNCTION_CODE, @L_PTR                   ; 0627
                                           6E   D6 000AC          INCL    L_PTR
                     00    BE              55   90 000AE          MOVB    OPERAND, @L_PTR                         ; 0628
                                           6E   D6 000B2          INCL    L_PTR
                     04    AE              03   C0 000B4          ADDL2   #3, L_LEN                               ; 0629
                           52              01   CE 000B8 5$:      MNEGL   #1, WHITE_SPACE                         ; 0634
                                           2D   11 000BB          BRB     10$                                     ; 0563
                           20              50   D1 000BD 6$:      CMPL    CH, #32                                 ; 0637
                                           09   13 000C0          BEQL    7$
                00000000G  8F              50   D1 000C2          CMPL    CH, #TAB
                                           13   12 000C9          BNEQ    8$
                FFFFFFFF   8F              52   D1 000CB 7$:      CMPL    WHITE_SPACE, #-1                        ; 0644
                                           16   12 000D2          BNEQ    10$
                           52        04    AE   D0 000D4          MOVL    L_LEN, WHITE_SPACE                      ; 0651
                     00    BE              20   90 000D8          MOVB    #32, @L_PTR                             ; 0652
                                           07   11 000DC          BRB     9$
                           52              01   CE 000DE 8$:      MNEGL   #1, WHITE_SPACE                         ; 0663
                     00    BE              50   90 000E1          MOVB    CH, @L_PTR                              ; 0664
                                           6E   D6 000E5 9$:      INCL    L_PTR
                                     04    AE   D6 000E7          INCL    L_LEN                                   ; 0665
    FF36   54              01        04    AC   F1 000EA 10$:     ACBL    SOURCE_LEN, #1, I, 1$                   ; 0555
                FFFFFFFF   8F              52   D1 000F1          CMPL    WHITE_SPACE, #-1                        ; 0676
                                           0C   13 000F8          BEQL    11$
          50               52        04    AE   C3 000FA          SUBL3   L_LEN, WHITE_SPACE, R0                  ; 0679
                           6E              50   C0 000FF          ADDL2   R0, L_PTR
                     04    AE              52   D0 00102          MOVL    WHITE_SPACE, L_LEN                      ; 0680
                     00    BE              2C   90 00106 11$:     MOVB    #44, @L_PTR                             ; 0683
```

```
                                    6E  D6 0010A         INCL    L_PTR                        ;
                 00   BE            20  90 0010C         MOVB    #32, @L_PTR                  ; 0684
                                    6E  D6 00110         INCL    L_PTR                        ;
                 04   AE            02  C0 00112         ADDL2   #2, L_LEN                    ; 0685
                      7E      0C    AC  7D 00116         MOVQ    PERMUTED_LEN, -(SP)          ; 0691
                              10    AE  9F 0011A         PUSHAB  LINE                         ;
                              10    AE  DD 0011D         PUSHL   L_LEN                        ;
                      66            04  FB 00120         CALLS   #4, FIX_CASE                 ;
                                    5E  DD 00123         PUSHL   SP                           ; 0696
                              08    AE  9F 00125         PUSHAB  L_LEN                        ;
                      7E      0C    AC  7D 00128         MOVQ    PERMUTED_LEN, -(SP)          ;
                      67            04  FB 0012C  12$:   CALLS   #4, COPY_STR                 ;
                      7E      14    AC  7D 0012F         MOVQ    XTN, -(SP)                   ; 0697
                              10    AE  9F 00133         PUSHAB  LINE                         ;
                              10    AE  DD 00136         PUSHL   L_LEN                        ;
          00000000G   EF            04  FB 00139         CALLS   #4, XOUT                     ;
                                        04 00140         RET                                  ; 0698

; Routine Size:  321 bytes,    Routine Base:  $CODE$ + 0144
```

```
;  534     0699  1 %SBTTL 'FIX_CASE -- Fix up case of permuted entry'
;  535     0700  1 ROUTINE FIX_CASE (B_LEN, B_PTR, P_LEN, P_PTR) : NOVALUE =
;  536     0701  1 !++
;  537     0702  1 !
;  538     0703  1 ! FUNCTIONAL DESCRIPTION:
;  539     0704  1 !
;  540     0705  1 !       This routine is called to fix up the case of the permuted index
;  541     0706  1 !       entry. The case of the first character of the new entry is raised.
;  542     0707  1 !       The case of the first character of the permuted text is lowered
;  543     0708  1 !       if there are any lower case characters in the first word of the
;  544     0709  1 !       permuted text.
;  545     0710  1 !
;  546     0711  1 ! FORMAL PARAMETERS:
;  547     0712  1 !
;  548     0713  1 !       B_LEN - Length of string which begins new entry
;  549     0714  1 !       B_PTR - Pointer to string which begins new entry
;  550     0715  1 !       P_LEN - Length of permuted string
;  551     0716  1 !       P_PTR - Pointer to permuted string
;  552     0717  1 !
;  553     0718  1 ! IMPLICIT INPUTS:
;  554     0719  1 !
;  555     0720  1 !       None
;  556     0721  1 !
;  557     0722  1 ! IMPLICIT OUTPUTS:
;  558     0723  1 !
;  559     0724  1 !       None
;  560     0725  1 !
;  561     0726  1 ! ROUTINE VALUE:
;  562     0727  1 ! COMPLETION CODES:
;  563     0728  1 !
;  564     0729  1 !       None
;  565     0730  1 !
;  566     0731  1 ! SIDE EFFECTS:
;  567     0732  1 !
;  568     0733  1 !       None
;  569     0734  1 !--
;  570     0735  2     BEGIN
;  571     0736  2     LOCAL
;  572     0737  2         PTR,
;  573     0738  2         LEN,
;  574     0739  2         FIRST_PTR,
;  575     0740  2         FIRST_CH,
;  576     0741  2         LOWER;
;  577     0742  2
;  578     0743  2     !
;  579     0744  2     ! Raise case of new first character if necessary
;  580     0745  2     !
;  581     0746  2     PTR = .B_PTR;
;  582     0747  2     LEN = .B_LEN;
;  583     0748  2
;  584     0749  2     INCR I FROM 1 TO .LEN DO
;  585     0750  3         BEGIN
;  586     0751  3         LOCAL
;  587     0752  3             CH;
;  588     0753  3
;  589     0754  3         CH = CH$RCHAR (.PTR);
;  590     0755  3
```

```
;  591          0756  3          IF .CH EQL RINTES
;  592          0757  3          THEN
;  593          0758  4              BEGIN
;  594          0759  4              !
;  595          0760  4              ! RUNOFF escape sequence - skip rest of sequence
;  596          0761  4              !
;  597          0762  4              CH$RCHAR_A (PTR);
;  598          0763  4              CH$RCHAR_A (PTR);
;  599          0764  4              CH$RCHAR_A (PTR);
;  600          0765  4              I = .I + 2;
;  601          0766  4              END
;  602          0767  3          ELSE
;  603          0768  4              BEGIN
;  604          0769  4              !
;  605          0770  4              ! Have first character in string
;  606          0771  4              !
;  607          0772  5              IF LOWER_LETTER (.CH)
;  608          0773  4              THEN
;  609          0774  4                  !
;  610          0775  4                  ! Character is a lowercase letter - convert to upper case
;  611          0776  4                  !
;  612          0777  4                  CH$WCHAR (UPPER_CASE (.CH), .PTR);
;  613          0778  4
;  614          0779  4              EXITLOOP;
;  615          0780  3              END;
;  616          0781  2          END;
;  617          0782  2
;  618          0783  2      !
;  619          0784  2      ! If there's no permuted text then just return
;  620          0785  2      !
;  621          0786  2      IF .P_LEN EQL 0 THEN RETURN;
;  622          0787  2
;  623          0788  2      !
;  624          0789  2      ! Find first character in permuted text
;  625          0790  2      !
;  626          0791  2      FIRST_PTR = .P_PTR;
;  627          0792  2
;  628          0793  2      DECR I FROM .P_LEN TO 1 DO
;  629          0794  3          BEGIN
;  630          0795  3
;  631          0796  3          FIRST_CH = CH$RCHAR (.FIRST_PTR);
;  632          0797  3
;  633          0798  3          IF .FIRST_CH EQL RINTES
;  634          0799  3          THEN
;  635          0800  4              BEGIN
;  636          0801  4              !
;  637          0802  4              ! RUNOFF escape sequence - skip it
;  638          0803  4              !
;  639          0804  4              CH$RCHAR_A (FIRST_PTR);
;  640          0805  4              CH$RCHAR_A (FIRST_PTR);
;  641          0806  4              CH$RCHAR_A (FIRST_PTR);
;  642          0807  4              I = .I - 2;
;  643          0808  4              END
;  644          0809  3          ELSE
;  645          0810  4              BEGIN
;  646          0811  4              !
;  647          0812  4              ! Have pointer to first character.
```

```
;  648     0813  4              ! Set length of remainder of string and exit.
;  649     0814  4              !
;  650     0815  4              LEN = .I - 1;
;  651     0816  4              EXITLOOP;
;  652     0817  3              END;
;  653     0818  2          END;
;  654     0819  2
;  655     0820  2
;  656     0821  3      IF UPPER_LETTER (.FIRST_CH)
;  657     0822  2      THEN
;  658     0823  3          BEGIN
;  659     0824  3          !
;  660     0825  3          ! Lower case of first character in permuted text if the
;  661     0826  3          ! second letter in the word is lowercase.
;  662     0827  3          !
;  663     0828  3          PTR = CH$PLUS (.FIRST_PTR, 1);
;  664     0829  3
;  665     0830  3          LOWER = FALSE;                          ! No lower case characters seen
;  666     0831  3
;  667     0832  3          INCR I FROM 1 TO .LEN DO
;  668     0833  4              BEGIN
;  669     0834  4              LOCAL
;  670     0835  4                  CH;
;  671     0836  4
;  672     0837  4              CH = CH$RCHAR_A (PTR);
;  673     0838  4
;  674     0839  4              SELECTONE TRUE OF
;  675     0840  4                  SET
;  676     0841  4
;  677     0842  4                  [.CH EQL RINTES]:
;  678     0843  5                      BEGIN
;  679     0844  5                      !
;  680     0845  5                      ! RUNOFF escape sequence - skip rest of sequence
;  681     0846  5                      !
;  682     0847  5                      CH$RCHAR_A (PTR);
;  683     0848  5                      CH$RCHAR_A (PTR);
;  684     0849  5                      I = .I + 2;
;  685     0850  4                      END;
;  686     0851  4
;  687     0852  4                  [(.CH EQL %C' ') OR (.CH EQL TAB)]:
;  688     0853  5                      BEGIN
;  689     0854  5                      !
;  690     0855  5                      ! Whitespace - end of word
;  691     0856  5                      ! Leave case alone.
;  692     0857  5                      !
;  693     0858  5                      EXITLOOP;
;  694     0859  4                      END;
;  695     0860  4
;  696     0861  4                  [LOWER_LETTER (.CH)]:
;  697     0862  5                      BEGIN
;  698     0863  5                      !
;  699     0864  5                      ! Lower case letter - note it and exit loop
;  700     0865  5                      !
;  701     0866  5                      LOWER = TRUE;
;  702     0867  5                      EXITLOOP;
;  703     0868  4                      END;
;  704     0869  4
```

```
4                       [UPPER_LETTER (.CH)]:
4                           !
4                           ! Second letter is upper case. Terminate scan but don't
4                           ! lower case of first letter.
4                           !
4                           EXITLOOP;
4                       [OTHERWISE]:
4                           !
4                           ! Some other character - ignore it
4                           !
4                           ;
4                       TES;
3                   END;
3
3               IF .LOWER THEN CH$WCHAR (LOWER_CASE (.FIRST_CH), .FIRST_PTR);
2               END;
2
1           END;
```

```
                                        07FC 00000 FIX_CASE:
                                                         .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10        ; 0700
                5A      00G  8F  9A 00002                MOVZBL  #RINTES, R10                            ;
                51      08   AC  D0 00006                MOVL    B_PTR, PTR                              ; 0746
                53      04   AC  D0 0000A                MOVL    B_LEN, LEN                              ; 0747
                             52  D4 0000E                CLRL    I                                       ; 0749
                             28  11 00010                BRB     3$                                      ;
                50           61  9A 00012 1$:            MOVZBL  (PTR), CH                               ; 0754
                5A           50  D1 00015                CMPL    CH, R10                                 ; 0756
                             08  12 00018                BNEQ    2$                                      ;
                51           03  C0 0001A                ADDL2   #3, PTR                                 ; 0764
                52           02  C0 0001D                ADDL2   #2, I                                   ; 0765
                             18  11 00020                BRB     3$                                      ; 0756
        00000061  8F         50  D1 00022 2$:            CMPL    CH, #97                                 ; 0772
                             13  19 00029                BLSS    4$                                      ;
        0000007A  8F         50  D1 0002B                CMPL    CH, #122                                ;
                             0A  14 00032                BGTR    4$                                      ;
     61         50           20  83 00034                SUBB3   #32, CH, (PTR)                          ; 0777
                             04  11 00038                BRB     4$                                      ; 0768
     D4         52           53  F3 0003A 3$:            AOBLEQ  LEN, I, 1$                              ; 0749
                        0C   AC  D5 0003E 4$:            TSTL    P_LEN                                   ; 0786
                             01  12 00041                BNEQ    5$                                      ;
                                 04 00043                RET                                             ;
                52      10   AC  D0 00044 5$:            MOVL    P_PTR, FIRST_PTR                        ; 0791
     50        0C   AC       01  C1 00048                ADDL3   #1, P_LEN, I                            ; 0793
                             16  11 0004D                BRB     8$                                      ;
                57           62  9A 0004F 6$:            MOVZBL  (FIRST_PTR), FIRST_CH                   ; 0796
                5A           57  D1 00052                CMPL    FIRST_CH, R10                           ; 0798
                             08  12 00055                BNEQ    7$                                      ;
                52           03  C0 00057                ADDL2   #3, FIRST_PTR                           ; 0806
                50           02  C2 0005A                SUBL2   #2, I                                   ; 0807
```

```
                           06  11 0005D        BRB     8$                              ; 0798
          53       FF      A0  9E 0005F 7$:    MOVAB   -1(R0), LEN                     ; 0815
                           03  11 00063        BRB     9$                              ; 0810
          E7               50  F5 00065 8$:    SOBGTR  I, 6$                           ; 0793
00000041  8F               57  D1 00068 9$:    CMPL    FIRST_CH, #65                   ; 0821
                           01  18 0006F        BGEQ    10$                             ;
                               04 00071        RET                                     ;
0000005A  8F               57  D1 00072 10$:   CMPL    FIRST_CH, #90                   ;
                           01  15 00079        BLEQ    11$                             ;
                               04 0007B        RET                                     ;
          51       01      A2  9E 0007C 11$:   MOVAB   1(R2), PTR                      ; 0828
                           58  D4 00080        CLRL    LOWER                           ; 0830
                           56  D4 00082        CLRL    I                               ; 0832
                           7D  11 00084        BRB     21$                             ;
          54               81  9A 00086 12$:   MOVZBL  (PTR)+, CH                      ; 0837
          5A               54  D1 00089        CMPL    CH, R10                         ; 0842
                           08  12 0008C        BNEQ    13$                             ;
          51               02  C0 0008E        ADDL2   #2, PTR                         ; 0848
          56               02  C0 00091        ADDL2   #2, I                           ; 0849
                           6D  11 00094        BRB     21$                             ; 0839
                           55  D4 00096 13$:   CLRL    R5                              ; 0852
          20               54  D1 00098        CMPL    CH, #32                         ;
                           02  12 0009B        BNEQ    14$                             ;
                           55  D6 0009D        INCL    R5                              ;
                           50  D4 0009F 14$:   CLRL    R0                              ;
00000000G 8F               54  D1 000A1        CMPL    CH, #TAB                        ;
                           02  12 000A8        BNEQ    15$                             ;
                           50  D6 000AA        INCL    R0                              ;
          50               55  C8 000AC 15$:   BISL2   R5, R0                          ;
          01               50  D1 000AF        CMPL    R0, #1                          ;
                           55  13 000B2        BEQL    22$                             ;
                           55  D4 000B4        CLRL    R5                              ; 0861
00000061  8F               54  D1 000B6        CMPL    CH, #97                         ;
                           02  19 000BD        BLSS    16$                             ;
                           55  D6 000BF        INCL    R5                              ;
                           50  D4 000C1 16$:   CLRL    R0                              ;
0000007A  8F               54  D1 000C3        CMPL    CH, #122                        ;
                           02  14 000CA        BGTR    17$                             ;
                           50  D6 000CC        INCL    R0                              ;
          59               55  D2 000CE 17$:   MCOML   R5, R9                          ;
          50               59  CA 000D1        BICL2   R9, R0                          ;
          01               50  D1 000D4        CMPL    R0, #1                          ;
                           05  12 000D7        BNEQ    18$                             ;
          58               01  D0 000D9        MOVL    #1, LOWER                       ; 0866
                           2B  11 000DC        BRB     22$                             ; 0862
                           55  D4 000DE 18$:   CLRL    R5                              ; 0870
00000041  8F               54  D1 000E0        CMPL    CH, #65                         ;
                           02  19 000E7        BLSS    19$                             ;
                           55  D6 000E9        INCL    R5                              ;
                           50  D4 000EB 19$:   CLRL    R0                              ;
0000005A  8F               54  D1 000ED        CMPL    CH, #90                         ;
                           02  14 000F4        BGTR    20$                             ;
                           50  D6 000F6        INCL    R0                              ;
          59               55  D2 000F8 20$:   MCOML   R5, R9                          ;
          50               59  CA 000FB        BICL2   R9, R0                          ;
          01               50  D1 000FE        CMPL    R0, #1                          ;
                           06  13 00101        BEQL    22$                             ;
```

```
                FF7D            56              01              53  F1 00103 21$:   ACBL    LEN, #1, I, 12$                 ; 0832
                                                04              58  E9 00109 22$:   BLBC    LOWER, 23$                      ; 0887
                                62              57              20  81 0010C        ADDB3   #32, FIRST_CH, (FIRST_PTR)      ;
                                                                    04 00110 23$:   RET                                     ; 0890

; Routine Size:  273 bytes,    Routine Base:  $CODE$ + 0285
```

```
;  727     0891  1 %SBTTL 'COPY_STR -- Copy string source to destination'
;  728     0892  1 ROUTINE COPY_STR (SOURCE_LEN, SOURCE_PTR, DEST_LEN, DEST_PTR) : NOVALUE =
;  729     0893  1 !++
;  730     0894  1 !
;  731     0895  1 ! FUNCTIONAL DESCRIPTION:
;  732     0896  1 !
;  733     0897  1 !       This routine copies the string described by SOURCE_LEN and
;  734     0898  1 !       SOURCE_PTR to the area pointed to by DEST_PTR.
;  735     0899  1 !
;  736     0900  1 !       Upon completion, DEST_PTR points to the next character position
;  737     0901  1 !       in the output string and DEST_LEN has been updated to reflect
;  738     0902  1 !       the number of characters copied.
;  739     0903  1 !
;  740     0904  1 !       Trailing white space is deleted from the source string.
;  741     0905  1 !
;  742     0906  1 ! FORMAL PARAMETERS:
;  743     0907  1 !
;  744     0908  1 !       SOURCE_LEN - Length of source string
;  745     0909  1 !       SOURCE_PTR - CH$PTR to source string
;  746     0910  1 !       DEST_LEN   - Address of destination string length variable
;  747     0911  1 !       DEST_PTR   - Address of a CH$PTR to destination string
;  748     0912  1 !
;  749     0913  1 ! IMPLICIT INPUTS:
;  750     0914  1 !
;  751     0915  1 !       None
;  752     0916  1 !
;  753     0917  1 ! IMPLICIT OUTPUTS:
;  754     0918  1 !
;  755     0919  1 !       None
;  756     0920  1 !
;  757     0921  1 ! ROUTINE VALUE:
;  758     0922  1 ! COMPLETION CODES:
;  759     0923  1 !
;  760     0924  1 !       None
;  761     0925  1 !
;  762     0926  1 ! SIDE EFFECTS:
;  763     0927  1 !
;  764     0928  1 !       None
;  765     0929  1 !--
;  766     0930  1
;  767     0931  2     BEGIN
;  768     0932  2
;  769     0933  2     BIND
;  770     0934  2         TO_LEN = .DEST_LEN,
;  771     0935  2         TO_PTR = .DEST_PTR;
;  772     0936  2
;  773     0937  2     LOCAL
;  774     0938  2         WHITE_SPACE,
;  775     0939  2         FROM_PTR;
;  776     0940  2
;  777     0941  2     FROM_PTR = .SOURCE_PTR;
;  778     0942  2     WHITE_SPACE = -1;
;  779     0943  2
;  780     0944  2     INCR I FROM 1 TO .SOURCE_LEN DO
;  781     0945  3         BEGIN
;  782     0946  3
;  783     0947  3         LOCAL
```

```
;  784     0948  3              CH;                                     ! Current character
;  785     0949  3
;  786     0950  3          CH = CH$RCHAR_A (FROM_PTR);
;  787     0951  3
;  788     0952  3          SELECTONE .CH OF
;  789     0953  3              SET
;  790     0954  3
;  791     0955  3              [RINTES]:
;  792     0956  4                  BEGIN
;  793     0957  4                  !
;  794     0958  4                  ! RUNOFF escape sequence
;  795     0959  4                  !
;  796     0960  4
;  797     0961  4                  LOCAL
;  798     0962  4                      FUNCTION_CODE,
;  799     0963  4                      OPERAND;
;  800     0964  4
;  801     0965  4                  FUNCTION_CODE = CH$RCHAR_A (FROM_PTR);
;  802     0966  4                  OPERAND = CH$RCHAR_A (FROM_PTR);
;  803     0967  4                  I = .I + 2;
;  804     0968  4
;  805     0969  4                  IF .FUNCTION_CODE NEQ %C'P'
;  806     0970  4                  THEN
;  807     0971  5                      BEGIN
;  808     0972  5                      !
;  809     0973  5                      ! Copy any runoff sequence except the nopermute sequence
;  810     0974  5                      !
;  811     0975  5                      CH$WCHAR_A (.CH, TO_PTR);
;  812     0976  5                      CH$WCHAR_A (.FUNCTION_CODE, TO_PTR);
;  813     0977  5                      CH$WCHAR_A (.OPERAND, TO_PTR);
;  814     0978  5                      TO_LEN = .TO_LEN + 3;
;  815     0979  4                      END;
;  816     0980  4
;  817     0981  4                  WHITE_SPACE = -1;                   ! No trailing white space
;  818     0982  3                  END;
;  819     0983  3
;  820     0984  3              [%C' ', TAB]:
;  821     0985  4                  BEGIN
;  822     0986  4                  !
;  823     0987  4                  ! Blank or TAB.
;  824     0988  4                  ! Remove extra whitespace by inserting a blank for
;  825     0989  4                  ! only the first one seen.
;  826     0990  4                  !
;  827     0991  4                  IF .WHITE_SPACE EQL -1
;  828     0992  4                  THEN
;  829     0993  5                      BEGIN
;  830     0994  5                      !
;  831     0995  5                      ! No blanks or tabs seen yet. Save position and insert a
;  832     0996  5                      ! blank in the output string.
;  833     0997  5                      !
;  834     0998  5                      WHITE_SPACE = .TO_LEN;
;  835     0999  5                      CH$WCHAR_A (%C' ', TO_PTR);
;  836     1000  5                      TO_LEN = .TO_LEN + 1;
;  837     1001  4                      END;
;  838     1002  4
;  839     1003  3                  END;
;  840     1004  3
```

```
;  3                 [OTHERWISE]:
;  4                     BEGIN
;  4                     !
;  4                     ! Copy a normal character
;  4                     !
;  4                     WHITE_SPACE = -1;
;  4                     CH$WCHAR_A (.CH, TO_PTR);
;  4                     TO_LEN = .TO_LEN + 1;
;  3                     END;
;  3
;  3                 TES;
;  3
;  2             END;
;  2
;  2         IF .WHITE_SPACE NEQ -1
;  2         THEN
;  3             BEGIN
;  3             !
;  3             ! Delete trailing whitespace
;  3             !
;  3             TO_PTR = CH$PLUS (.TO_PTR, (.WHITE_SPACE - .TO_LEN));
;  3             TO_LEN = .WHITE_SPACE;
;  2             END;
;  2
;  1         END;
```

```
                                 00FC 00000 COPY_STR:
                                                  .WORD   Save R2,R3,R4,R5,R6,R7                    ; 0892
               53      0C   AC   D0 00002         MOVL    DEST_LEN, R3                              ; 0934
               51      10   AC   D0 00006         MOVL    DEST_PTR, R1                              ; 0935
               55      08   AC   D0 0000A         MOVL    SOURCE_PTR, FROM_PTR                      ; 0941
               50           01   CE 0000E         MNEGL   #1, WHITE_SPACE                           ; 0942
                            54   D4 00011         CLRL    I                                         ; 0944
                            63   11 00013         BRB     7$                                        ;
               52           85   9A 00015 1$:     MOVZBL  (FROM_PTR)+, CH                           ; 0950
     00000000G 8F           52   D1 00018         CMPL    CH, #RINTES                               ; 0955
                            2C   12 0001F         BNEQ    3$                                        ;
               56           85   9A 00021         MOVZBL  (FROM_PTR)+, FUNCTION_CODE                ; 0965
               57           85   9A 00024         MOVZBL  (FROM_PTR)+, OPERAND                      ; 0966
               54           02   C0 00027         ADDL2   #2, I                                     ; 0967
     00000050  8F           56   D1 0002A         CMPL    FUNCTION_CODE, #80                        ; 0969
                            15   13 00031         BEQL    2$                                        ;
          00   B1           52   90 00033         MOVB    CH, @0(R1)                                ; 0975
                            61   D6 00037         INCL    (R1)                                      ;
          00   B1           56   90 00039         MOVB    FUNCTION_CODE, @0(R1)                     ; 0976
                            61   D6 0003D         INCL    (R1)                                      ;
          00   B1           57   90 0003F         MOVB    OPERAND, @0(R1)                           ; 0977
                            61   D6 00043         INCL    (R1)                                      ;
               63           03   C0 00045         ADDL2   #3, (R3)                                  ; 0978
               50           01   CE 00048 2$:     MNEGL   #1, WHITE_SPACE                           ; 0981
                            2B   11 0004B         BRB     7$                                        ; 0952
               20           52   D1 0004D 3$:     CMPL    CH, #32                                   ; 0984
                            09   13 00050         BEQL    4$                                        ;
```

```
       00000000G  8F      52  D1 00052        CMPL    CH, #TAB                     ;
                          12  12 00059        BNEQ    5$                           ;
       FFFFFFFF   8F      50  D1 0005B 4$:    CMPL    WHITE_SPACE, #-1             ; 0991
                          14  12 00062        BNEQ    7$                           ;
                  50      63  D0 00064        MOVL    (R3), WHITE_SPACE            ; 0998
            00    B1      20  90 00067        MOVB    #32, @0(R1)                  ; 0999
                          07  11 0006B        BRB     6$                           ;
                  50      01  CE 0006D 5$:    MNEGL   #1, WHITE_SPACE              ; 1010
            00    B1      52  90 00070        MOVB    CH, @0(R1)                   ; 1011
                          61  D6 00074 6$:    INCL    (R1)                         ;
                          63  D6 00076        INCL    (R3)                         ; 1012
    98            54  04  AC  F3 00078 7$:    AOBLEQ  SOURCE_LEN, I, 1$            ; 0944
       FFFFFFFF   8F      50  D1 0007D        CMPL    WHITE_SPACE, #-1             ; 1019
                          0A  13 00084        BEQL    8$                           ;
    52            50      63  C3 00086        SUBL3   (R3), WHITE_SPACE, R2        ; 1025
                  61      52  C0 0008A        ADDL2   R2, (R1)                     ;
                  63      50  D0 0008D        MOVL    WHITE_SPACE, (R3)            ; 1026
                              04 00090 8$:    RET                                  ; 1029

; Routine Size:  145 bytes,    Routine Base:  $CODE$ + 0396


;  866        1030  1
;  867        1031  1 END                                      ! End of module
;  868        1032  0 ELUDOM
```

```
;                              PSECT SUMMARY
;
;     Name                     Bytes                   Attributes
;
;  $CODE$                        1063  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                          Library Statistics
;
;                                     -------- Symbols --------      Pages      Processing
;     File                            Total    Loaded   Percent      Mapped     Time
;
;  _$255$DUA28:[SYSLIB]XPORT.L32;1      590        28         4       252        00:00.1


;                              COMMAND QUALIFIERS

;     BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:PERMUTE/OBJ=OBJ$:PERMUTE MSRC$:PERMUTE/UPDATE=(ENH$:PERMUTE)
```

```
; Size:               1063 code + 0 data bytes
; Run Time:             00:21.9
; Elapsed Time:         00:43.7
; Lines/CPU Min:         2824
; Lexemes/CPU-Min:      24725
; Memory Used:          133 pages
; Compilation Complete
```

PAGE
LIS

PERMUTE
LIS

PAGCMB
LIS

PANPA
LIS

RCS
LIS

POOL
LIS

RINIT
LIS

PERIOD
LIS

PUS
LIS

PUTNDX
LIS

PARSEP
LIS

PUTTOC
LIS

REQUIR
LIS

PAGMRG
LIS

PARAG
LIS

REPEAT
LIS