```
   SSSSSSSSSSS  UUU          UUU  MMM          MMM
   SSSSSSSSSSS  UUU          UUU  MMM          MMM
   SSSSSSSSSSS  UUU          UUU  MMM          MMM
SSS             UUU          UUU  MMMMMMM    MMMMMM
SSS             UUU          UUU  MMMMMMM    MMMMMM
SSS             UUU          UUU  MMMMMMM    MMMMMM
SSS             UUU          UUU  MMM MMM MMM MMM
SSS             UUU          UUU  MMM  MMM  MMM MMM
SSS             UUU          UUU  MMM   M   MMM MMM
   SSSSSSSSS    UUU          UUU  MMM       MMM MMM
   SSSSSSSSS    UUU          UUU  MMM       MMM MMM
   SSSSSSSSS    UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
SSSSSSSSSSS     UUUUUUUUUUUUUUUU  MMM       MMM MMM
SSSSSSSSSSS     UUUUUUUUUUUUUUUU  MMM       MMM MMM
SSSSSSSSSSS     UUUUUUUUUUUUUUUU  MMM       MMM MMM
SSSSSSSSSSS     UUUUUUUUUUUUUUUU  MMM       MMM MMM
```

```
  GGGGGGGG  EEEEEEEEEE  TTTTTTTTTT    CCCCCCCC  MM      MM  DDDDDDDD
  GGGGGGGG  EEEEEEEEEE  TTTTTTTTTT    CCCCCCCC  MM      MM  DDDDDDDD
GG          EE              TT       CC        MMMM  MMMM  DD      DD
GG          EE              TT       CC        MMMM  MMMM  DD      DD
GG          EE              TT       CC        MM  MM  MM  DD      DD
GG          EE              TT       CC        MM  MM  MM  DD      DD
GG          EEEEEEEE        TT       CC        MM      MM  DD      DD
GG          EEEEEEEE        TT       CC        MM      MM  DD      DD
GG  GGGGGG  EE              TT       CC        MM      MM  DD      DD
GG  GGGGGG  EE              TT       CC        MM      MM  DD      DD
GG      GG  EE              TT       CC        MM      MM  DD      DD      ....
GG      GG  EE              TT       CC        MM      MM  DD      DD      ....
  GGGGGG    EEEEEEEEEE      TT         CCCCCCCC  MM      MM  DDDDDDDD      ....
  GGGGGG    EEEEEEEEEE      TT         CCCCCCCC  MM      MM  DDDDDDDD      ....

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
```

```
0000     1          .TITLE  GETCMD
0000     2          .IDENT  'V04-000'
0000     3 ;
0000     4 ;*****************************************************************************
0000     5 ;*                                                                           *
0000     6 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     7 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     8 ;*  ALL RIGHTS RESERVED.                                                     *
0000     9 ;*                                                                           *
0000    10 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    11 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    12 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    13 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    14 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    15 ;*  TRANSFERRED.                                                             *
0000    16 ;*                                                                           *
0000    17 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    18 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    19 ;*  CORPORATION.                                                             *
0000    20 ;*                                                                           *
0000    21 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    22 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    23 ;*                                                                           *
0000    24 ;*                                                                           *
0000    25 ;*****************************************************************************
0000    26 ;
0000    27 ;
0000    28 ;++
0000    29 ;
0000    30 ; Facility:
0000    31 ;
0000    32 ;       SUMSLP utility
0000    33 ;
0000    34 ; Environment:
0000    35 ;
0000    36 ;       User mode
0000    37 ;
0000    38 ; Author:
0000    39 ;
0000    40 ;       R. Newland      18-Apr-1979
0000    41 ;
0000    42 ; Modified By:
0000    43 ;
0000    44 ;       V03-003 MTR0001         Mike Rhodes             27-Jul-1983
0000    45 ;               Correct processing of the /LISTING and /OUTPUT qualifiers.
0000    46 ;
0000    47 ;       V03-002 BLS0175         Benn Schreiber          16-JUN-1982
0000    48 ;               Use new CLI interface routines
0000    49 ;
0000    50 ;       V03-001 BLS0158         Benn Schreiber          16-Mar-1982
0000    51 ;               Remove $CLIDEFQUALEDIT, get values globally.
0000    52 ;--
```

```
                        .SBTTL  DECLARATIONS
                ;
                ;
                DEFSSLGEN                       ; Define SUMSLP general values
                $DSCDEF                         ; Descriptor offsets
                $FABDEF                         ; FAB block definitions
                DEFSSLFLG                       ; SUMSLP flag definitions
```

```
                                0000     62          .SBTTL  DATA
                           00000000     63          .PSECT  SUM$RO_DATA, NOEXE, NOWRT, LONG
                                0000     64
                                0000     65 INPUT_NAME:
                  54 55 50 4E 49 00'    0000     66          .ASCIC  /INPUT/
                                05      0000
                                0006     67 OUTPUT_NAME:
               54 55 50 54 55 4F 00'    0006     68          .ASCIC  /OUTPUT/
                                06      0006
                                000D     69 LISTING_NAME:
            47 4E 49 54 53 49 4C 00'    000D     70          .ASCIC  /LISTING/
                                07      000D
                                0015     71 UPDATE_NAME:
               45 54 41 44 50 55 00'    0015     72          .ASCIC  /UPDATE/
                                06      0015
                                001C     73 HEADER_NAME:
               52 45 44 41 45 48 00'    001C     74          .ASCIC  /HEADER/
                                06      001C
                                0023     75 LINE_NAME:
                  45 4E 49 4C 24 00'    0023     76          .ASCIC  /$LINE/
                                05      0023
                                0029     77 ;
                                0029     78 ;
                                0029     79 ;
```

```
                                0029    81          .SBTTL  GETCMD
                                0029    82 ;
                                0029    83 ; Subroutine to get command line information from the CLI and
                                0029    84 ; set up control variables appropriately
                                0029    85 ;
                                0029    86 ; Calling sequence:
                                0029    87 ;
                                0029    88 ;       BSB     SUM$GETCMD
                                0029    89 ;
                                0029    90 ; Inputs:
                                0029    91 ;
                                0029    92 ;       None
                                0029    93 ;
                                0029    94 ; Implicit inputs
                                0029    95 ;
                                0029    96 ;       The CLI data base
                                0029    97 ;
                                0029    98 ; Outputs:
                                0029    99 ;
                                0029   100 ;       xxx
                                0029   101 ;
                                0029   102 ; Implicit outputs
                                0029   103 ;
                                0029   104 ;       None
                                0029   105 ;
                                0029   106 ; Side effects:
                                0029   107 ;
                                0029   108 ;       None
                                0029   109 ;
                                0029   110 ;--
                                0029   111 ;
                           00000000   112          .PSECT  SUM$CODE, EXE, NOWRT
                                0000   113 ;
                                0000   114 SUM$GETCMD::
                         0C     BB  0000   115          PUSHR   #^M<R2,R3>
                         7E     7C  0002   116          CLRQ    -(SP)                           ;Create a descriptor
                      53     5E     D0  0004   117          MOVL    SP,R3
                   03 A3     02     90  0007   118          MOVB    #DSC$K_CLASS_D,DSC$B_CLASS(R3)
                                0008   119 ;
                                0008   120 ; Command line not printed anywhere, so this code not used
                                0008   121 ;
                                0008   122 ;       MOVAB   W^SUM$GQ_CMDLINE,R2             ;Initialize output descriptor
                                0008   123 ;       MOVB    #DSC$K_CLASS_D,DSC$B_CLASS(R2)  ;as dynamic
                                0008   124 ;       MOVAB   LINE_NAME,R0                    ;Item to get
                                0008   125 ;       BSBW    SUM_VALUE                       ;Get command line
                                0008   126 ;
                                0008   127 ; end of unused code
                                0008   128 ;
                   50 0000'CF     9E  0008   129          MOVAB   W^INPUT_NAME,R0                 ;Get input file spec
                      52     53     D0  000C   130          MOVL    R3,R2
                            0088     30  000F   131          BSBW    SUM_VALUE
                00B7'CF     00     FB  0012   132          CALLS   #0,@^INPUT_FILE
                                0017   133
                   50 0006'CF     9E  0017   134          MOVAB   W^OUTPUT_NAME,R0                ;Has an output file been
                            0068     30  001C   135          BSBW    SUM_PRESENT             ;requested?  By default one is
            00000000'8F     50     D1  001F   136          CMPL    R0, #CLI$_NEGATED       ;produced.  If not, then continue
                            14     13  0026   137          BEQL    10$                     ;parsing the command line.
```

```
00000000'EF   01   88  002C   138          BISB2   #SSL_M_OUTPUT, SUM$GL_FLAGS     ;Indicate output file requested.
     50  0006'CF   9E  0033   139          MOVAB   W^OUTPUT_NAME,R0                ;Get the file spec (if any) or
           0066   30  0038   140          BSBW    SUM_VALUE                       ;use the defaults from the NAM.
     00C0'CF   00   FB  003B   141          CALLS   #0,@^OUTPUT_FILE                ;Save the spec for later.
                    0040   142
     50  000D'CF   9E  0040   143 10$:     MOVAB   W^LISTING_NAME,R0               ;Has a listing been requested?
           0046   30  0045   144          BSBW    SUM_PRESENT                     ;If not, then continue parsing
         14 50    E9  0048   145          BLBC    R0,20$                          ;the command line.
00000000'EF   04   88  004B   146          BISB2   #SSL_M_LIST,SUM$GL_FLAGS         ;Indicate listing file requested.
     50  000D'CF   9E  0052   147          MOVAB   W^LISTING_NAME,R0               ;Get the file spec (if any) or
           0047   30  0057   148          BSBW    SUM_VALUE                       ;use the defaults from the NAM.
     00C9'CF   00   FB  005A   149          CALLS   #0,@^LIST_FILE                  ;Save the spec for later.
                    005F   150
     50  001C'CF   9E  005F   151 20$:     MOVAB   W^HEADER_NAME,R0                ;Test if /HEADER
           0027   30  0064   152          BSBW    SUM_PRESENT
         03 50    E9  0067   153          BLBC    R0,40$
           007E   30  006A   154          BSBW    HEAD_QUAL                       ; process /HEADER
                    006D   155
     50  0015'CF   9E  006D   156 40$:     MOVAB   W^UPDATE_NAME,R0                ;Test if /UPDATE
           0019   30  0072   157          BSBW    SUM_PRESENT
         07 50    E9  0075   158          BLBC    R0,60$
              53    DD  0078   159          PUSHL   R3                              ; Pass dynamic descriptor down
     0104'CF   01   FB  007A   160          CALLS   #1,W^UPDA_QUAL                  ; process /UPDATE
                    007F   161 ;
                    007F   162 ; Free dynamic string that we allocated
                    007F   163 ;
              53    DD  007F   164 60$:     PUSHL   R3                              ;Stack descriptor address
00000000'GF   01   FB  0081   165          CALLS   #1,G^STR$FREE1_DX
           5E   08   C0  0088   166          ADDL2   #8,SP                           ;Clear descriptor from stack
              0C   BA  008B   167          POPR    #^M<R2,R3>
                   05  008D   168          RSB
```

```
                                008E    170 ;
                                008E    171 ; Test if qualifier present
                                008E    172 ;
                                008E    173 ; Inputs:
                                008E    174 ;
                                008E    175 ;       R0      Pointer to ASCIC string name of entity
                                008E    176 ;
                                008E    177 ; Outputs:
                                008E    178 ;
                                008E    179 ;       R0      Return status from CLI$PRESENT
                                008E    180 ;
                                008E    181 SUM_PRESENT:
                          01 A0 9F      008E    182         PUSHAB  1(R0)                   ;Create descriptor on the stack
                       7E 60 9A         0091    183         MOVZBL  (R0),-(SP)              ;...
                          6E 9F         0094    184         PUSHAB  (SP)                    ;Stack descriptor address
                00000000'GF 01 FB       0096    185         CALLS   #1,G^CLI$PRESENT        ;Call CLI routine to test presence
                       5E 08 C0         009D    186         ADDL2   #8,SP                   ;Clear descriptor from stack
                          05            00A0    187         RSB                             ;Return with status in R0
                                00A1    188 ;
                                00A1    189 ; Get qualifier value
                                00A1    190 ;
                                00A1    191 ; Inputs:
                                00A1    192 ;
                                00A1    193 ;       R0      Pointer to ASCIC string name of entity
                                00A1    194 ;       R2      Pointer to output descriptor
                                00A1    195 ;
                                00A1    196 ; Outputs:
                                00A1    197 ;
                                00A1    198 ;       R0      Return status from CLI$GET_VALUE
                                00A1    199 ;       descriptor has value
                                00A1    200 ;
                                00A1    201 SUM_VALUE:
                          01 A0 9F      00A1    202         PUSHAB  1(R0)                   ;Create descriptor on the stack
                       7E 60 9A         00A4    203         MOVZBL  (R0),-(SP)
                          52 DD         00A7    204         PUSHL   R2                      ;Stack address of output descriptor
                          04 AE 9F      00A9    205         PUSHAB  4(SP)                   ; and address of name descriptor
                00000000'GF 02 FB       00AC    206         CALLS   #2,G^CLI$GET_VALUE      ;Call CLI to get value for qualifier
                       5E 08 C0         00B3    207         ADDL2   #8,SP                   ;Clear stack
                          05            00B6    208         RSB
```

```
                                00B7   210          .SBTTL  PARSE ACTIONS ROUTINES
                                00B7   211 ;
                                00B7   212 ; These routines are called when a parameter is present to extract
                                00B7   213 ; the file name descriptor and fill the appropriate FAB block.
                                00B7   214 ;
                                00B7   215 ;
                                00B7   216 INPUT_FILE:
                           003C 00B7   217          .WORD   ^M<R2,R3,R4,R5>
     52    0000'CF              9E 00B9   218          MOVAB   W^SUM$AX_INPUTFAB,R2    ; Get address of input FAB
                              10   11 00BE   219          BRB     FILE
                                00C0   220 ;
                                00C0   221 OUTPUT_FILE:
                           003C 00C0   222          .WORD   ^M<R2,R3,R4,R5>
     52    0000'CF              9E 00C2   223          MOVAB   W^SUM$AX_OUTPUFAB,R2    ; Get address of output FAB
                              07   11 00C7   224          BRB     FILE
                                00C9   225 ;
                                00C9   226 LIST_FILE:
                           003C 00C9   227          .WORD   ^M<R2,R3,R4,R5>
     52    0000'CF              9E 00CB   228          MOVAB   W^SUM$AX_LISTFAB,R2     ; Get address of list FAB
                                00D0   229 ;
                                00D0   230 FILE:
          7E    63              3C 00D0   231          MOVZWL  DSC$W_LENGTH(R3),-(SP)  ; Store length of block to allocate
        34 A2   6E              90 00D3   232          MOVB    (SP),FAB$B_FNS(R2)      ; Store length of spec in FAB
            2C  A2              9F 00D7   233          PUSHAB  FAB$L_FNA(R2)           ; Stack address to get allocated block
            04  AE              9F 00DA   234          PUSHAB  4(SP)                   ; Stack address of length of block
  00000000'GF   02              FB 00DD   235          CALLS   #2,G^LIB$GET_VM         ; Allocate memory for it
2C B2   04 B3   6E              28 00E4   236          MOVC3   (SP),@DSC$A_POINTER(R3),@FAB$L_FNA(R2) ; Copy spec in
                                04 00EA   237          RET
```

```
                                00EB   239          .SBTTL  QUALIFIER ACTION ROUTINES
                                00EB   240 ;
                                00EB   241 ; This routine is called if the /HEADER qualifier is seen
                                00EB   242 ;
                                00EB   243 ;
                                00EB   244 HEAD_QUAL:
00000000'EF    08    88         00EB   245          BISB2   #SSL_M_HEADER,SUM$GL_FLAGS
    001F'CF    03    90         00F2   246          MOVB    #FAB$C_VFC, -           ; Set record format to variable
                                00F7   247                  W^SUM$AX_OUTPUFAB+FAB$B_RFM ; with fixed control record
    003F'CF    0C    90         00F7   248          MOVB    #SSL$RHBSZE, -          ; Set output file FAB block to
                                00FC   249                  W^SUM$AX_OUTPUFAB+FAB$B_FSZ ; write record header buffer
0000'CF    0000'CF    9E         00FC   250          MOVAB   W^SUM$AX_RHB, -         ; and RAB block with record header
                                0103   251                  W^SUM$AX_OUTPURAB+RAB$L_RHB ; buffer address
                       05       0103   252          RSB
                                0104   253 ;
                                0104   254 ;
                                0104   255 ; This routine is called if the /UPDATE qualifier is called
                                0104   256 ;
                                0104   257 ;
                                0104   258 ; Inputs:
                                0104   259 ;
                                0104   260 ;       4(ap)   Address of scratch dynamic string descriptor
                                0104   261 ;
                                0104   262 UPDA_QUAL:
                          0000  0104   263          .WORD   0
      50    0015'CF    9E       0106   264          MOVAB   W^UPDATE_NAME,R0                ;Get ASCIC name address
              01 A0    9F       010B   265          PUSHAB  1(R0)                           ;Create descriptor
           7E    60    9A       010E   266          MOVZBL  (R0),-(SP)
           51    5E    D0       0111   267          MOVL    SP,R1                           ;Save descriptor address
             04 AC    DD       0114   268          PUSHL   4(AP)                           ;Pass a dynamic descriptor down
           0000'CF    9F       0117   269          PUSHAB  W^SUM$GL_UPDATES                ;Stack address of listhead
              51      DD       011B   270          PUSHL   R1                              ;Stack descriptor address
     00000000'GF      9F       011D   271          PUSHAB  G^CLI$GET_VALUE                 ;Address of routine to get value
00000000'GF    04    FB         0123   272          CALLS   #4,G^SUM$UPDATE_QUAL            ;Process the list
                       04       012A   273          RET
                                012B   274
                                012B   275          .END
```

```
BIT...                          = 00000005
CLI$GET_VALUE                     ********    X   03
CLI$PRESENT                       ********    X   03
CLI$_NEGATED                      ********    X   03
DSC$A_POINTER                   = 00000004
DSC$B_CLASS                     = 00000003
DSC$K_CLASS_D                   = 00000002
DSC$W_LENGTH                    = 00000000
FAB$B_FNS                       = 00000034
FAB$B_FSZ                       = 0000003F
FAB$B_RFM                       = 0000001F
FAB$C_VFC                       = 00000003
FAB$L_FNA                       = 0000002C
FILE                              000000D0  R       03
HEADER_NAME                       0000001C  R       02
HEAD_QUAL                         000000EB  R       03
INPUT_FILE                        000000B7  R       03
INPUT_NAME                        00000000  R       02
LIB$GET_VM                        ********      X   03
LINE_NAME                         00000023  R       02
LISTING_NAME                      0000000D  R       02
LIST_FILE                         000000C9  R       03
OUTPUT_FILE                       000000C0  R       03
OUTPUT_NAME                       00000006  R       02
RAB$L_RHB                         ********    X   03
SIZ...                          = 00000001
SSL$AULSZE                        00000018
SSL$AUTSZE                      = 00000010
SSL$BUFSZE                      = 000000FF
SSL$FILSZE                      = 00000040
SSL$LNOSZE                      = 00000006
SSL$RHBSZE                      = 0000000C
SSL_M_FORMFEED                  = 00000010
SSL_M_HEADER                    = 00000008
SSL_M_LIST                      = 00000004
SSL_M_MODE                      = 00000002
SSL_M_OUTPUT                    = 00000001
SSL_V_FORMFEED                  = 00000004
SSL_V_HEADER                    = 00000003
SSL_V_LIST                      = 00000002
SSL_V_MODE                      = 00000001
SSL_V_OUTPUT                    = 00000000
STR$FREE1_DX                      ********    X   03
SUM$AX_INPUTFAB                   ********    X   03
SUM$AX_LISTFAB                    ********    X   03
SUM$AX_OUTPUFAB                   ********    X   03
SUM$AX_OUTPURAB                   ********    X   03
SUM$AX_RHB                        ********    X   03
SUM$GETCMD                        00000000  RG      03
SUM$GL_FLAGS                      ********    X   03
SUM$GL_UPDATES                    ********    X   03
SUM$UPDATE_QUAL                   ********    X   03
SUM_PRESENT                       0000008F  R       03
SUM_VALUE                         000000A1  R       03
UPDATE_NAME                       00000015  R       02
UPDA_QUAL                         00000104  R       03
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
SUM$RO_DATA	00000029 (41.)	02 (2.)	NOPIC USR CON REL LCL NOSHR NOEXE RD NOWRT NOVEC LONG
SUM$CODE	0000012B (299.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	34	00:00:00.08	00:00:00.37
Command processing	111	00:00:00.57	00:00:01.57
Pass 1	192	00:00:03.69	00:00:10.63
Symbol table sort	0	00:00:00.46	00:00:00.97
Pass 2	64	00:00:00.92	00:00:02.62
Symbol table output	9	00:00:00.09	00:00:00.32
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	414	00:00:05.86	00:00:16.53

The working set limit was 1200 pages.
19897 bytes (39 pages) of virtual memory were used to buffer the intermediate code.
There were 20 pages of symbol table space allocated to hold 356 non-local and 4 local symbols.
275 source lines were read in Pass 1, producing 16 object records in Pass 2.
12 pages of virtual memory were used to define 11 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SUM.OBJ]SUM.MLB;1	2
_255DUA28:[SYSLIB]STARLET.MLB;2	6
TOTALS (all libraries)	8

399 GETS were required to define 8 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:GETCMD/OBJ=OBJ$:GETCMD MSRC$:GETCMD/UPDATE=(ENH$:GETCMD)+LIB$:SUM/LIB
```

SUMSLP
MAP

SRTTRNMSG
LIS

DATA
LIS

SUMSHR
MAP

SUMDATA
LIS

UTLCHKCLI
LIS

SUM

SUM
MAP

SUMDEF
MAR

MAIN
LIS

SUMED
LIS

SUMEDIT
LIS

GETCMD
LIS