

```

EEEEEEEEEE XX XX AAAAAA MM MM PPPPPPPP LL EEEEEEEEEE SSSSSSSS
EEEEEEEEEE XX XX AAAAAA MM MM PPPPPPPP LL EEEEEEEEEE SSSSSSSS
EEEEEEEEEE XX XX AAAAAA MM MM PPPPPPPP LL EEEEEEEEEE SSSSSSSS
EE XX XX AA AA MMMM MMMM PP PP LL EE SS
EE XX XX AA AA MMMM MMMM PP PP LL EE SS
EE XX XX AA AA MMMM MMMM PP PP LL EE SS
EE XX XX AA AA MM MM MM PP PP LL EE SS
EE XX XX AA AA MM MM MM PP PP LL EE SS
EE XX XX AA AA MM MM MM PP PP LL EE SS
EEEEEEEEEE XX XX AA AA MM MM PPPPPPPP LL EEEEEEEEEE SSSSSSS
EEEEEEEEEE XX XX AA AA MM MM PPPPPPPP LL EEEEEEEEEE SSSSSSS
EEEEEEEEEE XX XX AA AA MM MM PPPPPPPP LL EEEEEEEEEE SSSSSSS
EE XX XX AAAAAAAAAA MM MM PP LL EE SS
EE XX XX AAAAAAAAAA MM MM PP LL EE SS
EE XX XX AAAAAAAAAA MM MM PP LL EE SS
EE XX XX AA AA MM MM PP LL EE SS
EE XX XX AA AA MM MM PP LL EE SS
EE XX XX AA AA MM MM PP LL EE SS
EEEEEEEEEE XX XX AA AA MM MM PP LLLLLLLLLL EEEEEEEEEE SSSSSSSS
EEEEEEEEEE XX XX AA AA MM MM PP LLLLLLLLLL EEEEEEEEEE SSSSSSSS
EEEEEEEEEE XX XX AA AA MM MM PP LLLLLLLLLL EEEEEEEEEE SSSSSSSS

```

```

LL      BBBB8888      RRRRRRR      DDDDDDD      EEEEEEEEE      MM      MM      000000
LL      BBBB8888      RRRRRRR      DDDDDDD      EEEEEEEEE      MM      MM      000000
LL      BB      BB      RR      RR      DD      DD      EE      MMMM      MMMM      00      00
LL      BB      BB      RR      RR      DD      DD      EE      MMMM      MMMM      00      00
LL      BB      BB      RR      RR      DD      DD      EE      MM      MM      00      00
LL      BB      BB      RR      RR      DD      DD      EE      MM      MM      00      00
LL      BBBB8888      RRRRRRR      DD      DD      EE      MMM      MM      00      00
LL      BBBB8888      RRRRRRR      DD      DD      EE      MMM      MM      00      00
LL      BB      BB      RR      RR      DD      DD      EE      MM      MM      00      00
LL      BB      BB      RR      RR      DD      DD      EE      MM      MM      00      00
LL      BB      BB      RR      RR      DD      DD      EE      MM      MM      00      00
LL      BB      BB      RR      RR      DD      DD      EE      MM      MM      00      00
LL      BBBB8888      RR      RR      DDDDDDD      EEEEEEEEE      MM      MM      000000
LL      BBBB8888      RR      RR      DDDDDDD      EEEEEEEEE      MM      MM      000000

```

```

FFFFFFFF      000000      RRRRRRR
FFFFFFFF      000000      RRRRRRR
FF      00      00      RR      RR
FF      00      00      RR      RR
FF      00      00      RR      RR
FF      00      00      RR      RR
FFFFFFFF      00      00      RRRRRRR
FFFFFFFF      00      00      RRRRRRR
FF      00      00      RR      RR
FF      00      00      RR      RR
FF      00      00      RR      RR
FF      00      00      RR      RR
FF      000000      RR      RR
FF      000000      RR      RR

```

```
C      Demo program for library access procedures
C
C      Version 'V04-000'
```

```
C*****
C*
C* COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
C* DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
C* ALL RIGHTS RESERVED.
C*
C* THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
C* ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
C* INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
C* COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
C* OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
C* TRANSFERRED.
C*
C* THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
C* AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
C* CORPORATION.
C*
C* DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
C* SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
C*
C*****
```

```
C
C      PROGRAM LBR_DEMO
C
C      IMPLICIT INTEGER (A-Z)
C      EXTERNAL lib$get_input, list_module, lbr$_illcreopt
C
C      The common block is declared in a MACRO source module to gain
C      access to system definitions not available to FORTRAN.
C
C      COMMON /lbrdata/ func_read, func_create, func_update,
C      1 type_text, type_help, rmseof, class_dynamic, create_type,
C      2 create_keylen, create_alloc, create_idxmax, create_uhdmax,
C      3 create_entall
C
C      CHARACTER*128 library_name, library_rsn, module_name
C      CHARACTER*128 input_line
C
C      BYTE help_namblk (56), string_desc_bytes (8), dyn_desc_bytes (8)
C
C      DIMENSION create_options (0:49), old_module_rfa (2), module_rfa (2),
C      1 dyn_string (2), string_desc (2)
C
C      The equivalence of the STRING_DESC array with the STRING_DESC_BYTES
C      array is done to access the sstring descriptor class field.
C
C      EQUIVALENCE (string_desc, string_desc_bytes),
C      1 (dyn_string, dyn_desc_bytes)
C
C      library_open = .false.
```

```
      have_name = .false.
C
C      Initialize NAM block for use with HELP library
C
C      CALL nam_init (help_namblk, library_rsn)
C
C      Allocate a dynamic string and initialize string descriptors
C
      dyn_string (1) = 0
      dyn_string (2) = 0
      dyn_desc_bytes (4) = class_dynamic
      string_desc (1) = 0
      string_desc (2) = 0
      string_desc_bytes (4) = class_dynamic
      status =
10      1 lib$get1_dd (2048, string_desc)      !Allocate 2048 byte string
      IF (status) GOTO 100
      CALL lib$signal (%VAL (status))
      STOP
C
C      Main dispatch loop -- Get action and dispatch
C
100     TYPE 9000
      READ (5, *, END=300) action
      GOTO (1000, 2000, 3000, 4000, 5000, 6000, 7000), action + 1
      GOTO 100
200     CALL lib$signal (%VAL (status))
      GOTO 100
C
C      Close library and exit
C
300     IF (library_open) THEN
           status = lbr$close (library_index)
           IF (.NOT. status) GOTO 10
           END IF
      STOP
C
C      Give some help
C
1000    TYPE 9020
      GOTO 100
C
C      Name new library
C
C      If there is a library open, it will be closed. The new library name
C      is accepted.
C
2000    IF (library_open) THEN
           status = lbr$close (library_index)
           IF (.NOT. status) CALL lib$signal (%VAL (status))
           library_open = .false.
           END IF
      TYPE 9040
      READ (5, 9110, END=100) name_length, library_name
      library_name = library_name (1:name_length)/7'.TLB'
```

```

have_name = .true.
GOTO 100
C
C
C
3000 Open or Create a TEXT Library
IF (.NOT. have_name) GOTO 8000
TYPE 9540
READ (5, *, END = 100) create_flag
3100 IF (create_flag) THEN
TYPE 9060
READ (5, *, END=100) max_key_length
function = func_create
create_options (create_type) = type_text
create_options (create_keylen) = max_key_length
create_options (create_alloc) = 100
create_options (create_idxmax) = 1
create_options (create_uhdmax) = 0
create_options (create_entall) = 100
ELSE
function = func_update      !Opening existing library. Set for updates
END IF
C
C
C
Initialize librarian for this library
status = lbr$ini_control (library_index, function, type_text)
IF (.NOT. status) GOTO 200
C
C
C
Open or create the library
status = lbr$open (library_index, library_name, create_options)
C
C
C
If there is an illegal create option then it must be the value for the
maximum key length, so reprompt.
IF (status .EQ. %LOC (lbr$_illcreopt)) THEN
CALL lib$signal (lbr$_illcreopt)
GOTO 3100
END IF
IF (.NOT. status) GOTO 200
library_open = .TRUE.
GOTO 100
C
C
C
Insert or replace a module in the text library
4000 IF (.NOT. library_open) GOTO 8020
TYPE 9080
READ (5, 9110, END=100) name_length, module_name
replacing = lbr$lookup_key (library_index,
1 module_name (1:name_length), old_module_rfa)      !Determine if replacing or inserting module
TYPE 9100
4100 READ (5, 9110, END = 4200) line_length, input_line
status = lbr$put_record (library_index,
1 input_line (1:line_length), module_rfa)
4120 IF (.NOT. status) CALL lib$signal (%VAL (status))
GOTO 4100

```

```
C
C      Module text has been inserted into the library. Terminate the module
C
4200  status = lbr$put_end (library_index)
      IF (.NOT. status) CALL lib$signal (%VAL (status))
      status = lbr$replace_key (library_index, , !Insert or replace key
      1 module_name (1:name_length), old_module_rfa, module_rfa)
      IF (.NOT. status) CALL lib$signal (%VAL (status))
      status = .TRUE.
      IF (replacing) status = lbr$delete_data (library_index, !If replacing, delete old module
      1 old_module_rfa)
      IF (.NOT. status) GOTO 200
      GOTO 100

C
C      Extract module from library and type on terminal
C
5000  IF (.NOT. library_open) GOTO 8020
      TYPE 9400
      READ (5, 9050, END = 100) module_name
      status = lbr$lookup_key (library_index, module_name, module_rfa)
      IF (.NOT. status) GOTO 200
5100  status = lbr$get_record (library_index, string_desc, dyn_string)
      IF ((.NOT. status) .AND. status .NE. rmseof)
      1 CALL lib$signal (%VAL (status))
      IF (status .EQ. rmseof) GOTO 100
      CALL lib$put_output (dyn_string)
      GOTO 5100

C
C      List contents of index of SYSSHELP:HELPLIB.HLB
C
6000  status = lbr$ini_control (help_index, func_read, type_help)
      IF (.NOT. status) GOTO 200
      status = lbr$open (help_index, %DESCR ('SYSSHELP:HELPLIB.HLB'))
      IF (.NOT. status) GOTO 200
      status = lbr$get_index (help_index, 1, list_module)
      IF (.NOT. status) CALL lib$signal (%VAL (status))
      status = lbr$close (help_index)
      IF (.NOT. status) GOTO 200
      GOTO 100

C
C      Lookup help text in SYSSHELP:HELPLIB.HLB and display on the terminal
C
7000  TYPE 9200
      READ (5, 9110, END = 100) line_length, input_line
      status = lbr$output_help (lib$put_output, , input_line (1:line_length),
      1 %DESCR ('HELPLIB'), , lib$get_input)
      IF (.NOT. status) CALL lib$signal (%VAL (status))
      GOTO 100

C
C      Error routines
C

C
C      No library name given
C
8000  TYPE 9500
```

```
      GOTO 100
C
C      No library open
C
8020  TYPE 9520
      GOTO 100
C
C      Format statements
C
9000  FORMAT (' Action (0 for help): ', $)
9020  FORMAT (' Commands: ', /' 1 - Name library' , /,
1' 2 - Open or Create a text library' , /,
2' 3 - Replace/insert module' , /,
4' 4 - Extract module' , /,
3' 5 - List directory of SYSSHELP:HELPLIB.HLB' , /,
5' 6 - Lookup help text in SYSSHELP:HELPLIB.HLB')
9040  FORMAT (' New library name: ', $)
9050  FORMAT (A)
9060  FORMAT (' Maximum key length: ', $)
9080  FORMAT (' Module name: ', $)
9100  FORMAT (' Enter text. Terminate with a Control-Z:')
9110  FORMAT (Q, A)
9200  FORMAT (' Enter help keys: ', $)
9400  FORMAT (' Module to extract: ', $)
9500  FORMAT (' No library name given')
9520  FORMAT (' No library open')
9540  FORMAT
1 (' Open existing library (0) or create new library (1): ', $)
END

INTEGER FUNCTION list_module (keyname, keyrfa)

IMPLICIT INTEGER (A-Z)

CHARACTER *(*) keyname

TYPE *,keyname
list_module = .true.           !Return success
RETURN
END
```


XALINK
MAR

LABIOLINK
COM

DRMASTER
FOR

LPATEST
FOR

LABTOPEAK
FOR

LABIOSTR
COM

XAMESSAGE
MAR

XATEST
FOR

LABTOCOM
FOR

LABDEMO
COM

LABMBXDEF
FOR

LABIOSAMP
FOR

MAILCOMPRESS
COM

LABCHNDEF
FOR

CONNECT
COM

LABTOCON
FOR

LABDEMO
FOR

PEAK
FOR

DRCOPYBLD
COM

XIDRIVER
MAR

LABTOSEC
FOR

DRSLAVE
FOR

LABTOACO
FOR

LABTOCOMP
COM

LABIOSTAT
FOR

TESTLABIO
FOR