



CSI
INTERNATIONAL

This documentation applies to Release
5. of the program product BIM-EDIT.

Original Printing08/08/2003
Last Revised08/08/2003

TABLE OF CONTENTS

Chapter 1. Introduction.....	1
Chapter 2. Tutorial	2
Welcome to BIM-EDIT!	2
Logging On.....	2
Screen Layout	4
Creating Members	4
Editing Members	5
Positioning Commands.....	11
Ending the Edit Session.....	12
Logging Off.....	12
Onward and Upward.....	12
Chapter 3. General Information.....	13
Logging On and Logging Off	14
Screen Layout	15
Interactive Demo	16
Online Help	17
Command Syntax	18
Command Names	18
Operand Values (Positional Format)	18
Name=Value Operands (Keyword Format)	19
On/Off Operands	20
Operand Values in Quotes.....	20
Last Referenced Operands.....	21
Multiple Commands on a Line.....	22
Command Modifiers	22
Using PF keys for Commands	22
Command Syntax as it Relates to Procedures.....	23
The Log Area	24
Members and Libraries	25
Member Characteristics	25
Library Characteristics.....	25
Referencing Members and Libraries.....	26
Sessions.....	27
Starting Sessions.....	27
Ending Sessions.....	28
Groups of Sessions.....	28
Working on Multiple Sessions at Once	28
Positioning in a Session	29
Use of the LCA on LIBRARYx Sessions	30
The Stack Area	31
Editing.....	32
Sessions used for Editing.....	32
LCA Area Editing.....	32
Command Line Editing	33
Inter-User Mail	34
Access to Batch Processing	35
Presenting Members for Batch Processing.....	35
Accessing and Modifying Processing Results	35

TABLE OF CONTENTS

Job Completion Notification (VSE only)	36
Access to VSE Sublibraries and MVS PDSs.....	37
Security	38
Library Security	38
Member Security	38
POWER Job Entry and JES Data Set Security	39
Mail Security	39
VSE Sublibrary or MVS PDS Security	39
Command Security	39
Member Change Controls	40
Stamping	40
Auditing	40
Checkin/Checkout	40
Procedures	42
Variables.....	42
Procedure-Specific Commands	43
Batch Utility	44
Applications Interface	45
Customization.....	46
Command Customization	46
Other Customization.....	46
Chapter 4. Commands	47
User Command List.....	48
ALTER - Alter member attributes	54
ALTERL - Alter library attributes	57
ALTERP (VSE) - Alter POWER job entry attributes	58
ALTERP (MVS) - Alter JES job or data set attributes.....	61
ATTACH - Set library default for member access.....	64
ATTACHD (VSE) - Set VSE sublibrary default for access.....	65
ATTACHD (MVS) - Set PDS default for access.....	66
ATTACHX - Set user for mail access.....	67
AUDITCL - Delete member's audit trail to specified date.....	68
AUDITI - Write specified comment line to audit trail	70
AUDITRL - Back out member edits using audit trail	71
AUDITSM - Summarize member's audit trail.....	72
BACK - Position session display up one or more screens	74
BLANK - Alter session text - blank specified columns.....	75
BOTTOM - Position session display at down-most screen.....	76
CANCEL (MVS) - Terminate a JES job	77
CATAL (VSE) - Copy a member to VSE sublibrary.....	78
CATAL (MVS) - Copy a member to PDS	80
CENTER - Alter session text - center in specified columns.....	81
CHANGE - Alter session text - replace pattern with a string	82
CHECKASN - Move the working copy of a member to another user	86
CHECKIN - Replace a master member with its working copy	87
CHECKOUT - Create the working copy of a master member	88
CHECKPUR - Delete the working copy of a master member	90
COMPARE - Compare the text of two sessions.....	91
COMPILE - Submit a program compilation for batch processing.....	93
CONSOLEI - Write specified line to the operating system console	94

TABLE OF CONTENTS

COPY - Create copy of an existing member	95
COPYD (VSE) - Create copy of an existing VSE sublibrary member	97
COPYD (MVS) - Create copy of an existing PDS member	98
CURSOR - Set cursor to column set by last search command	99
DA (VSE) - Display operating system's processing status	100
DA (MVS) - Display operating system's processing status	102
DEFINE - Create new member	103
DEFINED (VSE) - Create new VSE sublibrary member	109
DEFINED (MVS) - Create new PDS member	111
DEFINEL - Create new library	112
DELETE - Alter session text - delete lines	113
DISCARD - End current session; delete of associated entry or message	114
DISPLAY - Display text of member, \$LOG, \$MAIL, or \$STACK	115
DISPLAYC (VSE) - Display System Console Messages	117
DISPLAYD (VSE) - Display text of VSE sublibrary member	120
DISPLAYD (MVS) - Display text of PDS member	122
DISPLAYS - Display a list of all active sessions	124
DOWN - Position session display down by one or more lines	125
DUP - Alter session text - duplicate current line	126
EDIT - Create edit session of an existing member	127
EDITD (VSE) - Create edit session of VSE sublibrary member	129
EDITD (MVS) - Create edit session of PDS member	130
END - End current session	131
EXECUTE - Process text of a member as commands	132
FALTER - Display / alter member attributes - formatted screen	134
FALTERL - Display / alter library attributes - formatted screen	136
FALTERP (VSE) - Display / alter POWER job entry attributes	137
FALTERP (MVS) - Display / alter JES job or data set attributes	139
FCOPY - Create copy of an existing member	141
FDEFINE - Create a new member - formatted screen	143
FDEFINEL - Create new library - formatted screen	145
FIND - Search session downward for pattern at a column	146
FINDUP - Like FIND, except search upward	148
FORMAT - Alter session text - wordwrap lines into paragraph	150
FORWARD - Position session display down one or more screens	152
FRENAME - Alter the name of a member	153
FSESSION - Alter session attributes - formatted screen	155
GET - Alter session text - insert text of member	156
GETD (VSE) - Alter session text - insert VSE sublibrary member	157
GETD (MVS) - Alter session text - insert PDS member	158
GETP (VSE) - Alter session text - insert text of POWER job entry	159
GETP (MVS) - Alter session text - insert text of JES data sets	162
GETQ - Alter session text - insert text of all incoming messages	165
GROUP - Switch between Session Groups	166
HELP - Display information about a command	167
HOLD (VSE) - Put POWER job entry on hold	169
HOLD (MVS) - Put JES job or data sets on hold	171
INCLUDE - Incorporate lines from a member during processing	173
IND\$FILE - PC File Transfer Interface	175
INQUIRE - Display member attributes	177
INQUIRED (VSE) - Display attributes of a VSE sublibrary member	178

TABLE OF CONTENTS

INQUIREP (VSE) - Display list of currently executing POWER job output	179
INQUIREP (MVS) - Display list of JES data sets for job	181
INSERTI - Alter session text - insert specified line after current	183
JOIN - Combine text of two lines	184
JUSTIFYL - Alter session text - left justify lines in a column range	186
JUSTIFYR - Alter session text - right justify lines in a column range	187
KEEP - Alter session text - blank all but a column range	188
KEYS - Display / alter function key settings - formatted screen	189
LADD - Alter session text - insert blank lines after current	191
LIBRARY - Display list of members in a library	192
LIBDS (VSE) - Display results of a LIBR SEARCH command	197
LIBRARYD (VSE) - Display list of VSE sublibrary members	199
LIBRARYD (MVS) - Display list of PDS members	201
LIBRARYL - Display list of libraries you can access	203
LIBRARYP (VSE) - Display list of POWER job entries	205
LIBRARYP (MVS) - Display list of JES jobs or data sets	207
LIBRARYQ - Display list of your mail messages	209
LIBRARYU - Display list of users	211
LIBSDL (VSE) - Display results of a LIBR LISTDIR SDL command	213
LIST - Display text of member, \$LOG, \$MAIL, or \$STACK	215
LISTD (VSE) - Display text of DOS/VSE sublibrary member	216
LISTD (MVS) - Display text of PDS member	218
LISTP (VSE) - Display text of POWER job entry	220
LISTP (MVS) - Display text of JES data sets	224
LOCATE - Search session downward to specified pattern	227
LOCATEU - Like LOCATE, except search upward	230
LOGI - Write specified line to \$LOG	233
LOGOFF - Exit from BIM-EDIT	234
LOWERCAS - Alter session text - translate lines to lower case	235
MAIL - Send message - member - to another user	236
MAILI - Send message - single line - to another user	238
MAILSESS - Send message - current session - to another user	240
MERGE - Alter session text - overlay non-blank \$STACK text	241
MESSAGE (VSE) - Display VSE Message Explanation	243
MOVE - Alter the name of or move a member	245
NEXT - Position session display down one or more lines	247
NFIND - Search down to line WITHOUT specified pattern at column	248
NFINDUP - Like NFIND, except search upward	250
OPEN - Display text of mail message	252
OVERLAY - Alter session text - put specified string at columns	253
PASSWORD - Alter user's own password	254
POSITION - Position session to a specified line number	255
PRINT - Print member	256
PROCESS - Process - SUBMIT, COMPILE or EXECUTE - a member	259
PROPAGAT - Alter session text - copy from columns to others	261
PURGE - Delete member	262
PURGED (VSE) - Delete DOS/VSE sublibrary member	264
PURGED (MVS) - Delete PDS member	265
PURGEL - Delete empty library	266
PURGE (VSE) - Delete POWER job entry	267
PURGE (MVS) - Delete JES job or data sets	269

TABLE OF CONTENTS

PURGEQ - Delete mail message.....	271
QUALIFY - Display all lines of a session having specified pattern.....	272
QUIT - End current session; retain any edits	276
REFRESH - Recreate display session	277
RELEASE (VSE) - Take POWER job entry off hold.....	278
RELEASE (MVS) - Take JES job or data sets off hold	280
RENAME - Alter the name of a member.....	282
RENAMED (VSE) - Alter the name of a DOS/VSE sublibrary member	284
RENAMED (MVS) - Alter the name of a PDS member.....	285
RESEQ - Alter session text - put sequence numbers at columns	286
RESET - Forget pending LCA bracket command	288
ROTATE - Select session to display.....	289
SAVE - End current session; retain any edits	291
SCAN - Search multiple members for a specified pattern.....	292
SCREEN - Set session display modes	297
SEARCH - Search session from top to specified pattern.....	305
SEPARATE - Alter session text - break paragraph into lines	307
SEQCHECK - Check the order of lines in the current session	309
SESS - Alter session attributes.....	311
SET - Set PF key commands or BIM-EDIT controls	313
SHIFT - Alter session text - move right or left in columns	316
SHOW - Display BIM-EDIT general or PF key information.....	318
SORT - Alter session text - reorder lines based on columns	319
SPLIT - Alter session text - break line at column or string	321
SQUEEZE - Alter session text - remove embedded blanks.....	323
STACK - Copy lines from current session to \$STACK	324
STACKI - Write a specified line to \$STACK	325
STAMPCL - Blank out a member's date stamps	326
SUBMIT (VSE) - Submit a member for batch processing	327
SUBMIT (MVS) - Submit a member for batch processing.....	329
SUBMITD (MVS) - Submit an MVS PDS member for batch processing	331
SWAP - Position session display to alternate marker	332
TOP - Position session display to up-most screen	333
UNDO - Back out audit for previous edit session.....	334
UP - Position session display up one or more lines	335
UPPERCAS - Alter session text - translate lines to upper case	337
VIEW - Position session display left or right	338
VTOC (VSE) - Create DASD VTOC display session	340
& (Ampersand) - Redisplay commands after processing.....	344
= (Equal Sign) - Process last command again	345
? (Question Mark) - Recall commands from the command stack	346
< (Less Than) - Select other logical screen	347
Chapter 5. LCA Commands	348
Repeat Count LCA Commands.....	349
Bracket LCA Commands	349
Transferring Lines	349
Multiple LCA Commands on a Screen	350
A(n) - Insert blank text lines after this text line	351
B(n) - Insert text lines from \$STACK before this text line	352
C(n),CC - Copy these text lines to \$STACK	353

TABLE OF CONTENTS

D(n),DD - Delete these text lines.....	354
E(n),EE - Edit this member	355
G - Merge text lines from \$STACK with these text lines	356
H(n),HH - Put these POWER job entries or JES jobs on hold	357
I(n) - Insert text lines from \$STACK after this text line.....	358
J(n),JJ - Merge text lines from \$STACK with these text lines	359
K(n),KK - Append these text lines to \$STACK	360
L(n),LL - Display text of this member, message, etc.	361
M(n),MM - Move - copy these text lines to \$STACK and delete	362
N(n),NN - Append these text lines to \$STACK and delete	363
P(n),PP - Delete these members, messages, etc.....	364
Q(n),QQ - Display / alter these members, libraries, etc.....	365
R(n),RR - Release - take these POWER job entries or JES jobs off hold	366
S(n),SS - Process this member - submit, compile or execute.....	367
T(n),TT - Display text of this member, etc.....	368
U(n),UU - Translate these text lines to upper case	369
W(n),WW - Translate these text lines to lower case	370
X(n),XX - Display list of JES or POWER datasets within a job	371
Y - Attach to this library	373
"(n)	374
- Position text line to the bottom of the screen.....	375
/ - Position session to this text line	376
<(n),<<(n) - Shift these text lines left	377
>(n),>>(n) - Shift these text lines right	378
+(n),++ - Center these text lines.....	379
(n,((- Left justify these text lines	380
)n,)) - Right justify these text lines.....	381
Chapter 6. Advanced Techniques	382
Extended Search Patterns	383
Extended Search Pattern Characters.....	383
Using Extended Search Pattern to Search for "Words".....	384
Using Extended Search Pattern "Wildcards".	385
Extended Search Pattern Combinations.....	385
Making Search Commands Work Like One Another	386
Using the % Extended Search Pattern Position Marker.....	386
Finding First Non-Blank in a text Line.....	387
Multiple Operations on a Screen	388
Using Sequences of Commands	390
Split-Screen Operation	392
Logical Tabbing	394
Hexadecimal Display and Editing	396
Displaying Text Hexadecimally	396
Editing Text Hexadecimally.....	397
Hexadecimal Translation at Processing Time.....	398
Member Stamping.....	399
Use of Stamping	399
Format of Stamp Entry	399
Administration of Stamping	399
Member Auditing.....	401
Use of Auditing.....	401

TABLE OF CONTENTS

Format of the Audit Trail.....	402
Administration of Auditing.....	404
Checkout/Checkin Facility	405
Use of Checkout/Checkin	405
INCLUDE Processing for Checked Out Members.....	406
Administration of Checkout/Checkin.....	406
Checkout/Checkin with Auditing and Stamping.....	407
Index	409

Chapter 1. Introduction

BIM-EDIT is a text storage and online editing system developed as a quality tool for data processing professionals. Our users found that it is also useful in their accounting, marketing, purchasing, and other departments.

BIM-EDIT provides:

- Library and member structure designed and optimized for text storage. You can create, rename, and purge libraries and members within the BIM-EDIT libraries. You can also directly access the common operating system program storage formats. You have a variety of methods to control access to members and libraries.
- Working environment. BIM-EDIT allows you to work on multiple activities at the same time and does not require that activities be closed to exit it. BIM-EDIT commands have a consistent format and virtually all commands are valid whatever activity is currently active.
- Full screen editor for altering the text of members. In addition to overtyping the text displayed on the screen, you can delete, move and copy blocks of lines from one location (or member) to another. You can change working location in a member in a variety of ways. You can overlay constant text, change some or all instances of one string of characters to another, shift text, and sort text. You can track and even "undo" changes made to a member.
- Access to batch processing. You can submit members to be processed in batch. You can display and control the output of your processing from your terminal.
- Integrated procedure processor. You can create your own BIM-EDIT commands. Conditional logic, parameter passing, and variable substitution provide additional versatility.
- Integrated support for REXX procedures.
- Inter-user mail system.

This manual is organized as follows:

- Chapter 2 is a tutorial designed for users who have not previously been exposed to a text editor.
- Chapter 3 provides general information about the system.
- Chapter 4 describes the commands.
- Chapter 5 describes LCA (line control area) commands.
- Chapter 6 describes some advanced techniques for using BIM-EDIT.

There is also a BIM-EDIT System Reference Manual, which describes the procedure language commands, using BIM-EDIT in batch, how to install BIM-EDIT, and how to customize BIM-EDIT.

Chapter 2. Tutorial

Welcome to BIM-EDIT!

This chapter is written to acquaint you with the basic functions of BIM-EDIT, assuming you haven't worked with a text editor before. If you are familiar with text editors, you may prefer to skip forward to Chapter 3, General Information. Another alternative to this chapter is to learn BIM-EDIT from the BIM-EDIT interactive DEMO system, which is described at the front of Chapter 3.

You will probably get the most out of this chapter if you actually perform the same BIM-EDIT operations presented as you encounter them in your reading. By the time you complete this chapter, you will be able to productively use BIM-EDIT to create and update text members.

Before you can start, you need to have been assigned a user ID and a password. If you haven't received these, contact your supervisor.

Now, let's get started.

Logging On

The first step is to get logged on. Logging on is the process of identifying to BIM-EDIT just who you are. You are also expected to enter your password along with your user ID. This allows BIM-EDIT to verify that you are, indeed, who you say you are.

Call up BIM-EDIT. This is done by clearing the screen, then typing in the BIM-EDIT transaction code as follows:

edtr

If that doesn't work, try entering it in upper case, as follows:

EDTR

If all goes well, you should see the following screen appear.

```
=> LOGON USER= _ ,PSWD=
-----1-----2-----3-----4-----5-----6-----7-----

BIM-EDIT RELEASE 5.6A
COPYRIGHT 1983, 2002
CONNECTIVITY SYSTEMS INC.
```

The cursor is positioned to allow input of your user id. After entering your user ID, tab to the password field. Enter your password. Note that the password field is secured, that is, you can't see what you're typing in. This is done to minimize the chance of someone else seeing your password.

Allow BIM-EDIT to process by pressing the ENTER key.

If BIM-EDIT doesn't like your logon, you'll see some sort of error message. If you can't figure out what's wrong, you'll need to get assistance before proceeding.

If the logon is successful, the following screen should appear.

```
=> _
-----1-----2-----3-----4-----5-----6-----7-----
CURRENT LIBRARY      : 4216                      SYSTEM : BIMEDIT
LAST REFERENCED MEMBER : (NULL)                   USER  : USR1
LAST REFERENCED JOB    : (NULL)                   TERMINAL: LL07
CURRENT VSE SUBLIBRARY : (NULL)                   DATE   : 02/01/2001
LAST REFERENCED VSE MBR: (NULL)                   TIME   : 19:32:30
CURRENT SESSION GROUP : A OF A
USED SESSIONS         : 0 OF 9

DO A "HELP XXXX" - XXXX BEING THE COMMAND YOU WANT TO SEE HELP FOR
ALL - LIST ALL COMMANDS          ED - EDIT A MEMBER
DQ  - SHOW QUEUES                LI - LIST A MEMBER
LIB - DISPLAY A LIBRARY          DISP - DISPLAY A MEMBER
ATT - ATTACH TO A LIBRARY        LCA - SHOWS LCA COMMANDS
LIBL - LIST AVAILABLE LIBRARIES  ALT - ALTER MEMBER ATTRIBUTES
ROT - ROTATE TO NEXT SESSION     DEFL - DEFINE A LIBRARY
GRP - GO TO NEXT GROUP OF SESSIONS DEF - DEFINE NEW MEMBER

BIM-EDIT RELEASE 5.6A
COPYRIGHT 1983, 2003
CONNECTIVITY SYSTEMS INC.
```

The "USR1" for the user and the "4216" for the current library will probably be different. The screen may look somewhat different if you use different computer system programs. More on this later.

Screen Layout

Let's take a look at the screen before going further.

The first line of the screen starts with "=>". You'll be entering commands just to the right of this arrow. In most cases your cursor will be positioned just to the right of the arrow, ready for command input.

The second line, currently blank, is used for informational purposes. If you enter an incorrect command, BIM-EDIT will respond with a message on this line. Also, later on, you'll see that when you are editing a member, information, such as the name of the member you are editing, will be displayed here.

The third line is the scale line. It serves two functions. First, if you have to enter text in a specific column, the scale will show you where to place the text. Secondly, it separates the command line and information line from the rest of the screen. You'll appreciate this when you begin editing members. If the current session is for a member that has logical tabbing active, the current tab character will be displayed on the scale line in each of the tab positions.

Creating members

Members are created with the DEFINE command.

When you define a member, you specify the member name, the member type, and potentially a whole list of other characteristics. The only data that must be entered is the member name, so that's all we're going to talk about here.

For additional information on DEFINE, or for that matter, any command presented in this chapter, you can refer to the detail command explanations given in Chapter 4, Commands.

Let's define a member called "MEMBER1". All you do is enter the following command:

```
=> define member1
```

BIM-EDIT will respond with a message saying that the member has been defined. (When BIM-EDIT shows a member name in a message, the member name will be prefixed with the library name, and a period. So, if your library is "4216", the message will indicate that member "4216.MEMBER1" has been defined.)

In this explanation, we assume that when your user ID was created, it was setup to work in its own library. The reason BIM-EDIT prefixes messages with the library name is that advanced users often work with multiple libraries -- it avoids confusion if the library name is always shown with the member name.

Editing Members

You are now ready to edit the member you just defined, member "MEMBER1". Editing is the process of adding, deleting, and updating text lines. Enter:

```
=> edit member1
```

A screen similar to the following should appear. Note that we continue to use "4216" as the library name, since the author doesn't know what your library is.

```
=>
EDIT  4216.MEMBER1                      SESS=A 1( 1) LINE=    0(    0)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF MEMBER --
      -- END OF MEMBER --
```

Look at the information line (line 2). It tells us four things:

- The type of session. A session is simply a unit of work. We refer to the process or environment of editing a member as an EDIT session. In this case, the type is EDIT. BIM-EDIT provides four types of sessions. We're only concerned about EDIT sessions in this chapter.
- The name of the member being edited.
- "SESS=A 1(1)" tells us that this session is the first session out of a total of one sessions in Group A. BIM-EDIT allows you to have multiple sessions and groups active at one time. This is an extremely useful feature. Its power will become apparent later. For now, you'll be working with just one session at a time.
- "LINE= 0(0)" tells us that we are currently positioned at line 0, and that there are a total of 0 lines in the member. Being positioned at line 0 means that you are positioned at the top of the member.

Notice the "-- TOP OF MEMBER --" and "-- END OF MEMBER --" lines. These lines are not real text lines of the member, but rather just serve to bracket the text lines.

Also notice the "*====*" characters on the "-- TOP OF MEMBER --" line. This area is known as the line control area, abbreviated throughout this manual as LCA.

The "-- TOP OF MEMBER --" line and all text lines of a member display this area.

The "-- END OF MEMBER --" line does not.

Within the LCA, we can enter commands that act upon that line and potentially following lines.

Now, let's get down to some real editing.

First, since we currently don't have any lines, let's create some. We will use the LCA A command. The LCA A command will add a specified number of blank lines immediately after the line where the A is entered.

Tab over to the LCA associated with the top of member line. Enter "a5" in the LCA, as follows:

```
=>
EDIT  4216.MEMBER1                      SESS=A 1( 1) LINE=      0(      0)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
a5==*  -- TOP OF MEMBER  --
      -- END OF MEMBER  --
```

Hit the ENTER key to allow BIM-EDIT to process the transaction.

BIM-EDIT will respond with the following screen.

```
=>
EDIT  4216.MEMBER1                      SESS=A 1( 1) LINE=      0(      5)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
***** -- TOP OF MEMBER  --
*****
*****
*****
*****
***** -- END OF MEMBER  --
```

Notice that 5 blank lines have been added. Look at "LINE= 0(5)" on the information line. This has been updated to indicate that there are now 5 lines in the member.

You don't have to enter a number along with the LCA A command. Had we just entered "a", a single blank line would have been added. Thus, "a" is equivalent to "a1".

Note the location of the cursor. BIM-EDIT made a judgement about what you wanted to do next. The judgement was made that since you added blank lines, you probably just wanted space to type new lines in. Thus, BIM-EDIT positions the cursor at the first blank line created. BIM-EDIT will do quite a bit of automatic cursor positioning. You'll get used to it very quickly.

You can overwrite anywhere in column 1 through 72 of lines displayed on the screen. That represents the space between the left hand side of the screen and the LCA on the right hand side.

Now, let's type some sort of data into these five blank lines. See if you can cause your screen to look like the following screen.

```
=>
EDIT  4216.MEMBER1                      SESS=A 1( 1) LINE=      0(      5)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF MEMBER --
*====* Have you ever studied history?
*====* Do you like math?
*====* Anyway, I like to read.
*====* Although, truthfully, it's hard to find the
*====* time.
*====* -- END OF MEMBER --
```

When your screen looks like this, allow BIM-EDIT to process it by pressing the ENTER key.

The screen that BIM-EDIT redisplay will look like this:

```
=>
EDIT  4216.MEMBER1                      SESS=A 1( 1) LINE=    0(    5)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF MEMBER --
*====* HAVE YOU EVER STUDIED HISTORY?
*====* DO YOU LIKE MATH?
*====* ANYWAY, I LIKE TO READ.
*====* ALTHOUGH, TRUTHFULLY, IT'S HARD TO FIND THE
*====* TIME.
*====* -- END OF MEMBER --
```

The lines have been translated to upper case. Whether lines are translated to upper case or left in mixed case is controlled by the CASE operand of the DEFINE command. You didn't specifically specify CASE when you defined the member, but BIM-EDIT set it to upper case by default. After a member has been defined, you can subsequently alter the CASE and other attributes with the ALTER command.

You can type over any existing line the same way you typed over a blank line. Just move your cursor to where you want to overwrite, and type away. Let's proceed further with LCA commands.

Suppose you want to take the group of 5 lines entered and duplicate it 10 times. That would give you a total of 55 lines - the original 5 plus an additional 50. The solution involves the use of two commands, the LCA C command and the LCA I command.

We will make use of an area known as \$STACK. \$STACK is simply a place to temporarily hold text lines. The LCA C command will copy lines into \$STACK. The LCA I command will retrieve lines from \$STACK and insert them in the member. (\$STACK tends to be an invisible area. You can look at it directly, but you seldom do.)

Enter the two LCA commands as follows:

```
=>
EDIT 4216.MEMBER1                      SESS=A 1( 1) LINE=    0(    5)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF MEMBER --
c5====* HAVE YOU EVER STUDIED HISTORY?
*====* DO YOU LIKE MATH?
*====* ANYWAY, I LIKE TO READ.
*====* ALTHOUGH, TRUTHFULLY, IT'S HARD TO FIND THE
i10====* TIME.
*====* -- END OF MEMBER --
```

Allow BIM-EDIT to process by pressing the ENTER key. BIM-EDIT will first process the "c5" command by clearing \$STACK, then copying the 5 lines into it. Then the "i10" command will insert the contents of \$STACK 10 times immediately after line #5.

BIM-EDIT will respond with the following result.

```
=>
EDIT 4216.MEMBER1                      SESS=A 1( 1) LINE=    0(   55)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF MEMBER --
*====* HAVE YOU EVER STUDIED HISTORY?
*====* DO YOU LIKE MATH?
*====* ANYWAY, I LIKE TO READ.
*====* ALTHOUGH, TRUTHFULLY, IT'S HARD TO FIND THE
*====* TIME.
*====* HAVE YOU EVER STUDIED HISTORY?
*====* DO YOU LIKE MATH?
*====* ANYWAY, I LIKE TO READ.
*====* ALTHOUGH, TRUTHFULLY, IT'S HARD TO FIND THE
*====* TIME.
*====* HAVE YOU EVER STUDIED HISTORY?
*====* DO YOU LIKE MATH?
*====* ANYWAY, I LIKE TO READ.
*====* ALTHOUGH, TRUTHFULLY, IT'S HARD TO FIND THE
*====* TIME.
*====* HAVE YOU EVER STUDIED HISTORY?
*====* DO YOU LIKE MATH?
*====* ANYWAY, I LIKE TO READ.
*====* ALTHOUGH, TRUTHFULLY, IT'S HARD TO FIND THE
*====* TIME.
```

We have a new situation - all of the lines held in the member cannot be displayed on the screen. In just a bit, we're going to explain the positioning commands. They'll allow you to move forward or backward within a member that cannot be wholly displayed on one screen. First, there's one more LCA command that is indispensable. And that is the LCA D command. The LCA D command allows you to delete specific lines. Suppose we really only intended to have 10 groups of our 5 lines, not 11 groups.

Let's delete the first five lines by entering the LCA D command as follows:

```
=>
EDIT  4216.MEMBER1                      SESS=A 1( 1) LINE=    0(   55)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF MEMBER --
d5====* HAVE YOU EVER STUDIED HISTORY?
*====* DO YOU LIKE MATH?
*====* ANYWAY, I LIKE TO READ.
*====* ALTHOUGH, TRUTHFULLY, IT'S HARD TO FIND THE
*====* TIME.
*====* HAVE YOU EVER STUDIED HISTORY?
*====* DO YOU LIKE MATH?
*====* ANYWAY, I LIKE TO READ.
*====* ALTHOUGH, TRUTHFULLY, IT'S HARD TO FIND THE
*====* TIME.
*====* HAVE YOU EVER STUDIED HISTORY?
*====* DO YOU LIKE MATH?
*====* ANYWAY, I LIKE TO READ.
*====* ALTHOUGH, TRUTHFULLY, IT'S HARD TO FIND THE
*====* TIME.
*====* HAVE YOU EVER STUDIED HISTORY?
*====* DO YOU LIKE MATH?
*====* ANYWAY, I LIKE TO READ.
*====* ALTHOUGH, TRUTHFULLY, IT'S HARD TO FIND THE
*====* TIME.
```

Allow BIM-EDIT to process by pressing the ENTER key. BIM-EDIT will respond as follows:

```
=>
EDIT  4216.MEMBER1                      SESS=A 1( 1) LINE=    0(   50)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF MEMBER --
*====* HAVE YOU EVER STUDIED HISTORY?
*====* DO YOU LIKE MATH?
*====* ANYWAY, I LIKE TO READ.
*====* ALTHOUGH, TRUTHFULLY, IT'S HARD TO FIND THE
*====* TIME.
*====* HAVE YOU EVER STUDIED HISTORY?
*====* DO YOU LIKE MATH?
*====* ANYWAY, I LIKE TO READ.
*====* ALTHOUGH, TRUTHFULLY, IT'S HARD TO FIND THE
*====* TIME.
*====* HAVE YOU EVER STUDIED HISTORY?
*====* DO YOU LIKE MATH?
*====* ANYWAY, I LIKE TO READ.
*====* ALTHOUGH, TRUTHFULLY, IT'S HARD TO FIND THE
*====* TIME.
*====* HAVE YOU EVER STUDIED HISTORY?
*====* DO YOU LIKE MATH?
*====* ANYWAY, I LIKE TO READ.
*====* ALTHOUGH, TRUTHFULLY, IT'S HARD TO FIND THE
*====* TIME.
```

Notice that the "LINE= 0(50)" indicates that the member now contains only 50 lines.

There are quite a few more LCA commands, but the most important ones you've learned. For more information on LCA commands, refer to Chapter 5, LCA.

Positioning Commands

Now let's look at the most common positioning commands. These commands are entered on the first screen line, just beyond the "=>" arrow. We're all done showing examples, so go ahead and try these commands as they are presented.

You can position forward an entire screen by the FORWARD command (abbreviated FO). Simply enter:

```
=> fo
```

To position backward an entire screen, use the BACK command (abbreviated BA) as follows:

```
=> ba
```

You can position forward a specified number of lines by the NEXT command (abbreviated N). To position forward 6 lines, enter:

```
=> n 6
```

The UP command (abbreviated U) works the same way, but positions backward. To position backward 12 lines, enter:

```
=> u 12
```

For both NEXT and UP, if you just want to position one line, just enter the command. You don't have to enter the "1".

There will be times when you want to go right to the top or bottom of the member.

Use TOP (abbreviated T) to position at the top of the member. Simply enter:

```
=> t
```

Use BOTTOM (abbreviated B) to position at the bottom of the member. Simply enter:

```
=> b
```

You may want to position to a particular line number. Use POSITION (abbreviated P) for this. To position to line number 25, enter:

```
=> p 25
```

You now have the means of positioning anywhere within a member, regardless of how large the member is.

Ending the Edit Session

Now let's end the EDIT session. This is accomplished by the SAVE command. Simply enter:

```
=> save
```

When you edit the member again, you will find the text just as it was at the time of the SAVE command.

Logging Off

When you're done with BIM-EDIT, for now, you'll want to log off. This is accomplished by entering:

```
=> logoff
```

Onward and Upward

BIM-EDIT has much, much more to offer than what we've presented here. Gradually you'll want to expose yourself to the more advanced features.

Chapter 3 provides general background information on BIM-EDIT. Chapter 4 describes the commands available from the command line. Chapter 5 describes the LCA commands. Chapter 6 describes some advanced techniques.

Chapter 3. General Information

This chapter discusses topics of general interest to a BIM-EDIT user. It is written with the assumption that you have worked with a text editor before. If you have not had this experience, you should start with Chapter 2, Tutorial, use BIM-EDIT a little, and then return to this chapter. The following are the sections of this chapter:

- Logging On and Logging Off
- Screen Layout
- Interactive Demo
- Online Help
- Command Syntax
- The Log Area
- Members and Libraries
- Sessions
- The Stack Area
- Editing
- Inter-User Mail
- Access to Batch Processing
- Access to VSE Sublibraries and MVS PDSs
- Security
- Member Change Controls
- Procedures
- Batch Utility
- Applications Interface
- Customization

This chapter discusses many topics briefly. You needn't try to remember all the details presented here, just use them to form an overall understanding of BIM-EDIT. The details are repeated (more precisely and with examples) in the reference chapters.

Logging On and Logging Off

How you invoke online BIM-EDIT depends on how your site has installed it.

If you access BIM-EDIT from CICS, it can be invoked in one of two ways:

- Enter the transaction code (usually EDTR), the user id, and the password in one command. If your user id is AB01 and your password is XM30, you would enter the following:

```
edtr ab01,xm30
```

For security reasons, your System Administrator may disable this method.

- Enter just the transaction code. BIM-EDIT displays a formatted screen that requests your user ID and password. The password field will not display as you type it, so that others can't see your password.

On VSE, if you access BIM-EDIT through the BTAM interface, enter the transaction code. BIM-EDIT displays a screen for your user ID and password.

If you access BIM-EDIT from VTAM, request BIM-EDIT from the VTAM initial menu screen. BIM-EDIT displays a screen that requests your user ID and password.

On MVS, if you access BIM-EDIT from TSO, enter a statement similar to:

```
call 'bimedit.loadlib(bimtsed)'
```

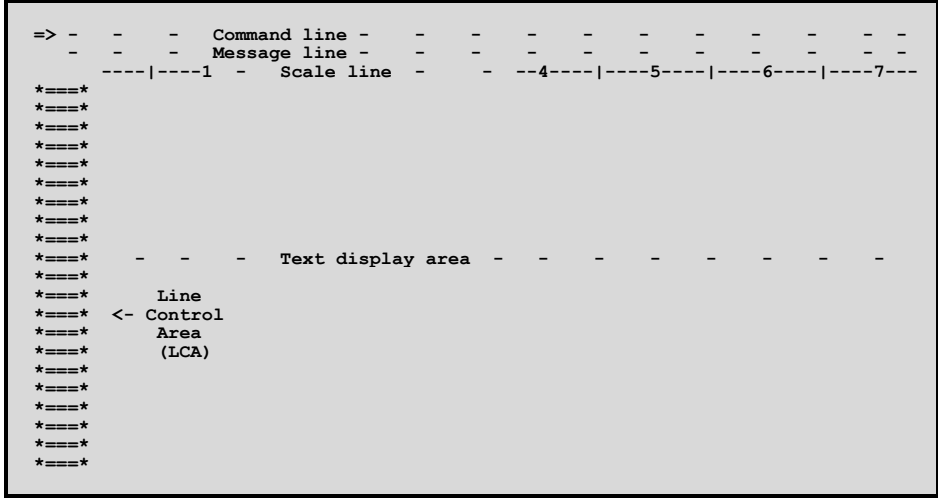
BIM-EDIT will display a screen that requests your user ID and password.

(Logon may work differently than this if your site has customized it.)

Your System Administrator can tell you the proper logon procedure.

Logging off of BIM-EDIT is as simple as entering the LOGOFF command. You do not have to close your current activities to log off. When you next logon, the screen and environment will be exactly the same as it was immediately before your LOGOFF. Activities will be preserved in open status and settings such as PF key commands and currently attached library are also preserved.

Unless the facility has been disabled at your site, you can change your password using the PASSWORD command.



Command Line	Commands are entered on this line.
Message Line	Displays information and messages.
Scale Line	A scale line shows column alignment and viewing range.
Text Display Area	This area if used for displaying text lines.
Line Control Area	(LCA) The five left- or right-most columns on certain displays. Text manipulation commands entered here consist of a single letter which affects the line they were entered on (for instance, the LCA command D deletes the line it was placed on).

The above screen layout is the default one. The `SCREEN` command can be used to change:

- The position and presence of the LCA.
- The number of text lines displayed before the scale line.
- Whether hexadecimal display is provided.
- Whether the screen is to be split into multiple editing areas.
- The character size of the screen.

Interactive Demo

BIM-EDIT incorporates an interactive demonstration facility which you can use to explore the product as a whole or to explore specific commands. The DEMO facility operates as a procedure within BIM-EDIT and is set up so that you can actually try out virtually all of the commands within DEMO's environment.

In order to access DEMO

1. You must be authorized by your site's System Administrator to use DEMO. While you are authorized to use DEMO, your BIM-EDIT will be a little less efficient and some additional BIM-EDIT library space will be consumed. When you don't expect to use the DEMO facility for several days, you may reclaim these resources by asking your System Administrator to de-authorize you. The procedures for the System Administrator are described in the BIM-EDIT System Reference Manual.
2. You use DEMO by entering the command DEMO. When you enter the command, DEMO will save your entire current status (open sessions, \$STACK contents, and so forth). The more sessions you have open and the larger these sessions are, the longer it will take to enter DEMO.
3. When you have finished using DEMO, exit it by following the instructions it displays. DEMO will restore your status at entry. How long it takes depends on how many sessions of what size you had open.

(DEMO SAVES your open EDIT sessions when you enter it and re-EDITs the members when you exit. It recreates other sessions by executing their associated commands. These implicit SAVES and REFRESHes may have implications for your processing if you use DEMO when you have sessions open.)

Online Help

The BIM-EDIT online help facility is accessed using the HELP command.

At all times, the HELP command is available for immediate access to the reference manual descriptions of the commands, the LCA commands, and the predefined variables. HELP is followed by the name of the topic to display. By simply entering HELP without a topic, you can get a list of the topics available.

Example of HELP display

```
=>
DISP -> help change                                SESS=A 3( 3) LINE=    0( 136)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----|-----
-- TOP OF DISPLAY --
Use CHANGE to replace occurrences of a specified string of current session text
characters with another specified string. Occurrences will be replaced in a
specified column range for a specified number of lines starting with the
current line.

CHANGE may also be entered as CH or C.

-----|-----
|                                     Required Operands                                     |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| OSTR   | is the "old string" to be replaced.                                         | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| NSTR   | is the "new" replacement string.  Wherever the OSTR string is |
|         | found, NSTR will replace it.                                                    |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|

-----|-----
|                                     Optional Operands                                     |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| FCT    | specifies the number of lines to apply the change to.  FCT may |
|         | also be specified with an asterisk(*), which indicates change to |
|         | end of session.  If FCT is not specified, only the current line |
|         | will be changed.                                                    | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| ZONE   | is the column range in which the search and replacement will |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
```

Command Syntax

BIM-EDIT has very consistent rules for command entry. If you attempt to enter commands which do not follow the rules, BIM-EDIT will reject them.

All commands consist of two parts: the "command name" (for example HELP) and zero or more "operands" (for example, in the case of HELP, the name of the topic for which help is desired). The command name specifies what action BIM-EDIT is to perform. The operands specify what is to be acted upon or how the action is to be performed.

Command Names

Command names are single words. They can be entered in upper case, lower case, or a combination. Frequently-used commands have optional abbreviated names, often one or two characters long. The command name must be separated from any operands by at least one space. Different commands have different operands.

Operand Values (Positional Format)

Operands are usually entered as a sequence of words or numbers following the command name. Each word or number is an operand value. For example, to request help on the LOCATE command, you could enter

```
=> help locate
```

HELP is the command name and "locate" is the value of its operand (the topic on which help is desired).

If there are several operands, they are entered in sequence after the command name, separated by a comma (,), spaces, or both. For example

```
=> define member-1, cobol
```

"member-1" is the value of the first operand of the DEFINE command (the name of the member to be created) and "cobol" is the value of the second operand (the type of member to create). When operand values are entered one after the other like this, the format is called "positional" format in contrast to the "keyword" format described below.

A particular operand is either "required" (meaning the command will be rejected if it is omitted) or "optional" (meaning it won't). In the example of the DEFINE command above, the member name is required, because it wouldn't make sense to create a member without specifying a name for it. But the member type is optional -- if it isn't specified, a default type will be used.

If the value of an operand is text (for example, the name or type of a member) BIM-EDIT will translate it to upper case letters if that is appropriate. In general, you do not need to worry about the capitalization of text operands.

Name=Value Operands (Keyword Format)

Entering operand values in sequence can be awkward when a command has many optional operands. You have to use the command's operand table to determine how many consecutive commas to use to "skip over" operands you don't want to enter. For example, if you wanted to enter a DEFINE command with the value "u" for the operand which specifies that the member you are creating is to have only upper case text (the sixth of twelve operands), you could enter:

```
=> define member6,,,,,u
```

There is an easier way. Every operand has an "operand name" which can be used to specify which value is being entered. The operand name is followed by an equal sign (=) and the value for that operand. Since CASE is the name of the operand for the DEFINE command which was entered above, you would enter:

```
=> define member6 case=u
```

This is both easier to type and easier to remember, since care has been taken to assure that all references to a particular feature use the same operand name for it. The name=value format is called the "keyword" format.

When you use keyword format, you can enter operands in any order that suits you. Positional and keyword formats can be mixed on a single command (as the example above demonstrates) with the restriction that any keyword operands must follow all positional operands.

As a final example on this topic, the following commands are equivalent:

```
=> define member6,cobol,,,,u
=> define member6 cobol case=u
=> define mem=member6 type=cobol case=u
=> define type=cobol case=u mem=member6
```

The function, abbreviated names, and operands of every command are described in Chapter 4, Commands, in this manual, and Chapter 3, Advanced User Commands, and Chapter 8, Operator and Administrator Commands, in the System Reference Manual. The same text is displayed by HELP. In the descriptions, a table is provided of each command's operands. This table gives the operand names and the operand sequence BIM-EDIT expects. It specifies whether each operand is required or optional and what action is taken when an optional operand is omitted.

On/Off Operands

Whenever an operand is described as having the values "ON" or "OFF", you can also use "YES" or "NO", "Y" or "N", and "1" or "0".

Operand Values in Quotes

If an operand value contains any of the characters single quote ('), double quote ("), comma (,), space (), equal sign (=), or semicolon (;), it must be enclosed in either single or double quotes to be acceptable to BIM-EDIT. Except in the procedures V commands, any operand value may be enclosed in quotes without any alteration in meaning (in practice, one encloses any long text operand value in quotes as a matter of course).

The following would be an example of entering an operand value in quotes:

```
=> define member5,title="this is member 5"
```

Operand values can be enclosed in quotes whether positional or keyword format of entry is used.

If an operand value contains quote characters, it can be enclosed in the other (single or double) type of quotes or the enclosed quote characters can be doubled. For example:

```
title="St. Mary's Cathedral"  
title='St. Mary' 's Cathedral '  
title='The "new" version'
```

"Last Referenced" Operands

You can make use of "last referenced member" whenever you execute a series of commands against the same member, as in create-edit, save-process or display-purge sequences. If you omit the name of a member, BIM-EDIT will assume you wish to process the member you last referenced.

On VSE, you can make similar use of "last referenced POWER job entry" and "last referenced VSE sublibrary member".

On MVS, you can make similar use of "last referenced JES data sets" and "last referenced MVS PDS member".

To see how much typing you can save, consider the following sequence:

=> define mymember	Creates a member MYMEMBER in the current library. Sets "last referenced member".
=> edit	No member name specified, edits "last referenced member" (MYMEMBER)
=> save	Completes edit, sets "last referenced member" (in case you used other member reference commands while you were editing)
=> process	No member name specified, processes "last referenced member" (MYMEMBER). Sets "last referenced POWER job entry" or "last referenced JES data set".
=> listp	No POWER job entry or JES data set specified, displays the output of the previous process command.

Note that the member name was only typed once and the POWER job entry or JES data set were never typed.

(PURGE is a special case: to reduce the possibility of carelessly deleting a member, you must use "*" for the name to delete the "last referenced member".)

The current "last referenced ..." entities are displayed on the home screen.

Multiple Commands on a Line

You can enter multiple commands on a command line by separating them with semicolons (;). For example, to create and then edit a MEMBER2, you could type

```
=> define member2;edit
```

The rules for multiple commands are:

- Commands are processed left to right.
- If an error is detected, processing ends.
- The screen displayed at end will be the result of all commands successfully processed.

Command Modifiers

Four special characters provide additional command facilities:

1. Prefix a command with ampersand (&) to cause it to be redisplayed on the command line when BIM-EDIT responds to the enter key. By hitting the enter key again, you can then execute the command again, or you can change part of the command before hitting the enter key. A common example is repeatedly searching for a word:

```
=> &locate word
```

2. Enter the equal sign (=) to cause the last command to be reprocessed. This can be used with formatting commands such as FORMAT or SEPARATE which need to be executed in several places on a screen.
3. Enter the question mark (?) to cause the last command to be redisplayed. This can be used if you enter a command without an ampersand prefix and then realize that your next command will be almost the same. You then use question mark to display it, change it and then hit enter to re-process.
4. Enter the less than sign (<) to switch your commands to process against the text in the other screen when you are in split screen mode.

Using PF keys for Commands

PF keys in BIM-EDIT are fast ways to invoke commands. The SET or KEYS commands assign any command (or a sequence of commands, using the semicolon separator) to a PF key. The SHOW PF command displays what commands your keys are set to. Whenever you press the PF key, the command(s) will be executed. The commands you set will be remembered by BIM-EDIT until you change them.

Command Syntax as it Relates to Procedures

As will be discussed later in this chapter, you can develop your own "commands" using BIM-EDIT's procedure facility. A procedure is simply a member containing commands. Procedures behave to the user exactly as if they were a command (infact, some of the "commands" supplied with BIM-EDIT are procedures). The same syntax rules are used for procedure names and procedure parameters as are used for command names and operands.

All commands (and in fact other procedures) can be invoked inside a procedure. There is only the following (useful) syntax difference when commands are used inside a procedure: only one command may appear per line -- text to the right of a semicolon (;) will be ignored by BIM-EDIT and can be used for comments.

The Log Area

Each user has a "log area" (\$LOG) which automatically traces the results of certain commands. For example, when a member is purged, a "member purged" line is written to \$LOG. Detailed updates to the text of a member, however, are not logged. (Updates of this sort are handled by the audit facility. See "Member Change Controls" later in this chapter.)

In addition to lines that are automatically logged, lines can be explicitly logged by the LOGF and LOGI commands.

\$LOG can be viewed at any time by entering:

```
=> list $log
```

which will generate a display like this:

```
=>
LIST $LOG                                SESS=A 3( 3) LINE= 0( 502)
-----1-----2-----3-----4-----5-----6-----7--
*==* -- TOP OF $LOG --
*==* ## "BIMTEXTP" SUBMITTED, NUMBER = 670, ON 06/02/1996 AT 12:14:36 ##
*==* ## MEMBER "ED51A.BIADTODO" SAVED ##
*==* ## "BIMTEXTP" SUBMITTED, NUMBER = 671, ON 06/02/1996 AT 12:15:24 ##
*==* ## LOGOFF COMPLETE:  DATE=06/02/1996, TIME=18:29:13, TERM=LL01 ##
*==* =====
*==* ## LOGON COMPLETE:   DATE=06/03/1996, TIME=12:16:17, TERM=LL05 ##
*==* ## LOGOFF COMPLETE:  DATE=06/03/1996, TIME=18:43:14, TERM=LL05 ##
*==* =====
*==* ## LOGON COMPLETE:   DATE=06/03/1996, TIME=18:52:15, TERM=DA2Y ##
*==* ## LOGOFF COMPLETE:  DATE=06/03/1996, TIME=18:55:05, TERM=DA2Y ##
*==* =====
*==* ## LOGON COMPLETE:   DATE=06/06/1996, TIME=10:14:17, TERM=LL01 ##
*==* ## "BIMTEXTP" SUBMITTED, NUMBER = 977, ON 06/06/1996 AT 10:20:04 ##
*==* ## MEMBER "BIM.BIZUSRF-PARTIAL" DEFINED ##
*==* ## MEMBER "BIM.BIZUSRF-PARTIAL" SAVED ##
*==* ## MESSAGE MAILED TO USER "GJB" ##
*==* ## MEMBER "BIM.BIZUSRF-PARTIAL" SAVED ##
*==* ## LOGOFF COMPLETE:  DATE=06/06/1996, TIME=19:17:26, TERM=LL01 ##
*==* =====
*==* ## LOGON COMPLETE:   DATE=06/07/1996, TIME=09:43:58, TERM=LL01 ##
*==* ## MESSAGE "GJB -> MWD 72 06/07/1996 16:35:41" PURGED ##
*==* ## MEMBER "ED51A.BIADTODO" ALTERED ##
*==* ## MEMBER "ED51A.BIADTODO" ALTERED ##
```

This display can be useful if you can't remember what name you gave a member, when you last submitted a member for processing or what an error message really said.

When your \$LOG reaches the maximum number of lines specified by your System Administrator, the oldest lines will be reused for new lines. The most recent lines are always at the bottom.

Members and Libraries

BIM-EDIT stores text in units called "members", which are in turn grouped into "libraries". In an office equipment analogy, members are like file folders and a library is like the file drawer that contains the folders.

Member Characteristics

A member is simply a collection of lines of text. It can contain anything you need to store or work on: a program, a memo, a procedure, or even a database. It can contain up to 999,999 lines of text, unless your System Administrator has specified a lower limit. Each line can be up to 253 columns wide.

Members are created with the DEFINE or COPY commands. They can be deleted with the PURGE command and renamed using the RENAME command. You can display a list of your members using the LIBRARY command.

In addition to text, members have "attributes" which determine how BIM-EDIT works with them. These are:

TITLE, USER	descriptive information for library displays
ATTR, LIBDEF, TYPE	how to process the member when submitted
AUDIT, STAMP, CHECK	change control values for member editing
CASE, NULLS	how to treat characters entered in the member
SEQ, ZONE, FCOL	column values for column dependent editing TCOL column values for tabbing
PSWD	password protection against unauthorized access

Additional values which are kept include the date, time and by whom the member was created and the date, time and by whom it was last edited.

Member attributes can be specified on the DEFINE command and can be changed with the ALTER and FALTER commands. The INQUIRE command can be used to display them.

When an EDIT session is started, member attributes are used to set session attributes. Session attributes can be changed while editing using the SESS and FSESS commands.

Rather than specifying several attributes when members are created, common collections of attributes are attached to particular "member types". The types are often customized for a particular site.

Library Characteristics

Libraries can contain any number of members. Libraries typically are used to group members by topic or by ownership. Member access security is based on library grouping. (See "Security" later in this chapter.)

A library must be created with the `DEFINEL` command before you can put members in it. An empty library (one with no members) can be deleted with the `PURGEL` command. You can display the libraries you can access with the `LIBRARYL` command.

Libraries have only two attributes: `TITLE` and `USER`. These can be specified on the `DEFINEL` command and changed with the `ALTERL` and `FALTERL` commands.

Referencing Members and Libraries

Members are referenced by name. A member name can be up to 16 characters in length, composed of the letters A-Z, the numerals 0-9, and the special characters dash (-), dollar sign (\$), pound sign (#), and the underscore (_). Member names cannot have a period (.) in them unless you have set the system variable `MMPEXTNM` to either '1' or '2'.

While BIM-EDIT supports the underscore (_) in the member name, it is not supported by all of the subsystems that BIM-EDIT interfaces with (for example, `POWER`). Care should be used when deciding which member names may contain the underscore.

A reference to a member includes or implies a reference to a library. A library name can be up to 16 characters in length, composed of the letters A-Z, the numerals 0-9, and the special characters period (.), dash (-), dollar sign (\$), and the underscore (_). (In other words, it is like a member name except a library name can contain a period.)

Members can be referenced in one of three ways.

1. Specify just the member name. BIM-EDIT assumes the member is in the currently attached library. (This library is set by the `ATTACH` command and will remain set until you change it.) For example, if the currently attached library is `OM11` and you want to edit the member `OMREXIO` in library `OM11`, you enter:

```
=> edit omrexio
```

If your BIM-EDIT system has system variable `MMPEXTNM` set to '1', and the member name you are referencing contains a period in its name, you must include a '\' as the first character:

```
=> \data.txt
```

If you omit the '\', BIM-EDIT will assume that 'data' is the library and 'txt' is the member name. If system variable `MMPEXTNM` has been set to '2', the leading '\' is not required.

2. Specify the library and member name, connected by either a period or a '\'. If the currently attached library is `$SYS` and you want to edit the member `OMREXIO` in library `OM11`, you enter:

```
=> edit om11.omrexio      or  => edit om11\omrexio
```

Note that in this way, you can reference any member, at any time, in any library you have access to, whether you are attached or not.

3. Omit the member name entirely. In this way you will obtain the "last referenced member". This is discussed in more detail in the "Command Syntax" section earlier in this chapter.

Access to members is restricted as described under "Security" later in this chapter.

Sessions

To change or look at text in BIM-EDIT, you must create a "session". A session is simply an area to work or view in. (If a member is considered to be like a file folder in a library file drawer, a session is like a file folder open on your desk. However, members are not the only source of sessions.) Up to 99 sessions per user may be active concurrently. The maximum number of sessions allowed can be limited by user via the DEFINEU and ALTERU commands.

Starting Sessions

Sessions are created by entering certain commands. For example, the command "edit ompgetp" will create an EDIT session of the member OMPGETP. Sessions are held on a chain. When a session is created, it is added to the chain after the current session and becomes the current session.

Here are the six types of sessions and the commands that create them:

DISP	Many commands create display sessions. The common ones are: LIBRARYx Displays lists of entities (members, libraries, POWER job entries, JES data sets, VSE sublibrary members, MVS PDS members, messages). QUALIFY Displays text lines in the current session matching a specified text search value. SCAN Displays text lines in all members in specified libraries matching a specified text search value. SHOW Displays BIM-EDIT general information and PF key settings. Since display sessions are often static displays of changing situations, the REFRESH command is provided to end and re-create DISP sessions quickly.
EDIT	These sessions are used to edit members. EDIT sessions are created by the EDIT and EDITD commands. Editing is discussed in detail in "Editing" later in this chapter.
LIST	These sessions display the text of members, \$LOG, \$MAIL, or \$STACK. LIST sessions are created by the LIST and LISTD commands. LIST sessions are like EDIT sessions except that you cannot change the text area and more than one user can simultaneously access the same member.
LISTD	These sessions display the text of VSE sublibrary members or MVS PDS members. LISTD sessions are created by the LISTD command and behave like LIST sessions.
LISTP	These sessions display the text of POWER job entries or MVS data sets. LISTP sessions are created by the LISTP command and behave like LIST sessions.

MAIL These sessions display the text of messages. MAIL sessions are created by the OPEN command and behave like LIST sessions.

Ending Sessions

A session remains active until terminated by the DISCARD, END or SAVE commands. These commands terminate the session that is current when they are issued. When a session is terminated, the previous session in the chain becomes the current session. If there are no more sessions, the home screen is displayed.

Groups of Sessions

Managing upwards of 99 active sessions can be confusing. To make this easier, BIM-EDIT supports a concept called 'Groups'. You can arrange your sessions into up to nine groups identified by the letters 'A' to 'I' which appears immediately following the SESS= on the second line of the screen. For example, "SESS=B 1(1)". Each group can contain any number of sessions as long as the sum of all sessions in all groups does not exceed 99 or the limit established for your user id.

The GROUP command is used to switch from one group to another. The result of the GROUP command is to assign a new 'active' group. BIM-EDIT always adds newly started sessions to the 'active' group. When you switch to a group that has no active sessions, the BIM-EDIT home screen will be displayed. The home screen contains a summary of the count of active groups and sessions, and indicates which group is the 'active' group. The maximum number of groups allowed can be limited by user via the DEFINEU and ALTERU commands.

Working on Multiple Sessions at Once

By issuing various forms of the ROTATE and GROUP commands, any active session in any group can be viewed. The message line SESS= field displays the current group, session number and, in parentheses, the total number of active sessions in the current group. For example, "SESS=A 1(4)" denotes that the current session is the first of four sessions in group A.

The SHOW command can be used to display a summary of the count of active sessions in each group.

The SCREEN command (SPLIT option) can be used to display two logical screens at once. Using the ROTATE command, these logical screens can show different sessions to compare two members or a member and its output listing. This method can only be used to view two sessions in the same group.

Positioning in a Session

You can set your viewing position in the current session (whatever type it may be) using positioning commands, which determine the session line which will display on the "current line", just below the scale line.

You can position to a specific line. The commands BOTTOM, POSITION, and TOP go to absolute line locations in the member. The commands BACK, DOWN, FORWARD, NEXT, and UP move relative to the current position. These commands are often assigned to PF keys. The LCA slash (/) command marks the new current line.

You can also position by line contents. The commands FIND, FINDUP, LOCATE, LOCATEU, NFOUND, and NFOUNDUP go to the next line matching or not matching a specified text search value in specified columns. A text search value can be simply a string of characters or an Extended Search Pattern requesting more complex tests. Used after one of the positioning commands, the CURSOR command will place the cursor at the instance which was matched.

Each session contains two line position markers. The SWAP command can be used to switch between these markers as if they were bookmarks.

The VIEW command can be used to position the session display horizontally when working on sessions which contain data beyond the column width of your screen.

Use of the LCA on LIBRARYx Sessions

A special series of LCA commands are available to manage entities directly from LIBRARYx sessions. If you place one of these on the same line as an entity, the action indicated will be performed to that entity:

E	Edit a member	Q	Display/alter attributes
H	Put POWER job entry on hold, put JES job/data sets on hold	R	Take POWER job entry off hold, take JES job/data sets off hold
L	Display text of entity	S	Process member
P	Purge entity	Y	Attach to library

LCA commands are described in detail in Chapter 5, LCA Commands.

The Stack Area

Each user has a text line holding area known as the stack area (\$STACK). \$STACK is used to hold lines the same way a member holds lines. You can copy lines to \$STACK, and you can retrieve lines from \$STACK.

Text from any type of session can be copied to \$STACK with the STACK command and the LCA commands C, CC, K, or KK. In particular, lines can be copied from a read-only LIST, LISTD or LISTP session into an EDIT session using \$STACK as intermediary. The LCA commands M, MM, N, and NN write lines to \$STACK but since they also delete them from the session, they can only be used in an EDIT or DISPLAY session. The commands STACKF, which can only be used from a procedure, and STACKI can be used to write specific lines to \$STACK.

The GET command can insert \$STACK lines into the current session. You would enter:

```
=> get $stack
```

The LCA I and B commands perform a similar function.

You can see the contents of \$STACK by entering:

```
=> list $stack
```

Editing

In BIM-EDIT, you can edit (change the text of) EDIT and DISP sessions in three ways:

- Overtyping the text display area.
- LCA commands. These single letter commands are entered next to the line they affect. They are used frequently to add, delete, copy and move lines.
- Command line commands. These are used for large scale changes, repetitive changes, and when editing is done within a procedure.

BIM-EDIT provides special features to display and change member text in hexadecimal and to allow a character to function as if it were a "tab". These features are described in Chapter 6, Advanced Techniques.

The commands that change session position (see "Sessions" earlier in this chapter) are often used while editing.

Sessions used for Editing

Most editing uses EDIT sessions started with the EDIT or EDITD commands.

An EDIT session is a working copy of a member. The changes you make are not stored in the member until you terminate the session with the SAVE command. If you desire, you can discard the effect of an edit session by terminating it with the END command with the NOSAVE option. Retrievals from a member that is undergoing editing will obtain the unchanged text of the member. (Some references, such as starting another EDIT session of a member, will be rejected when the member is being edited.)

Changes to session text are saved to disk on every screen transmission. Since sessions will be retained in open status if anything happens to BIM-EDIT, it is not necessary to periodically terminate sessions to reduce risk of loss. (BIM-EDIT has been designed to protect your entry investment to the greatest degree possible, including from power outages.) Sessions can literally be left open for years without problems; however it is sometimes desirable to end a session so that the END NOSAVE command can be used to recover from a truly disastrous editing error without also losing many desired changes.

DISP sessions are rarely edited. The feature is provided for convenience when using LCA commands with LIBRARYx sessions. When a DISP session is terminated, the changes are lost.

LCA Area Editing

LCA commands operate on the line where they are entered (for instance, the LCA command D deletes the line it is placed on). Many LCA commands can be followed by digits to indicate how many times they should be repeated (D5 to delete 5 lines) or doubled to bracket a series of lines they should be performed against (DD on the first and last lines of a series to be deleted). The command line RESET command can be used to forget a pending LCA bracket command.

LCA commands are used most frequently to delete or add lines. "D" deletes lines, "A" adds blank lines, "" duplicates lines. "I" and "B" insert the contents of \$STACK. "C" and "K" copy lines to \$STACK. "M" and "N" copy lines to \$STACK and then delete them.

LCA commands can also be used to modify specific columns or shift data within text lines. "J" and "G" place characters from \$STACK into blank columns. "+" centers text. "(" moves text to the left as far as possible. ")" moves text to the right as far as possible. "<" moves text to the left a specified number of columns. ">" moves text to the right a specified number of columns.

Two LCA commands perform translation. "W" translates letters to lower case. "U" translates letters to upper case.

LCA commands are described in detail in Chapter 5, LCA Commands.

Command Line Editing

Editing commands operate on the lines from the current line (the line where the session is positioned) downward for a specified number of lines. It is common to use the LCA "/" command to reposition the session prior to an editing command. If the number of lines is not specified, only the current line is affected. Many commands limit their effects to a particular column range.

The editing commands which have LCA equivalents are generally less convenient to use than the LCA version. Thus they tend to be used for special purposes such as editing from a procedure or using multiple commands on the command line. Commands of this type: DELETE deletes lines (LCA "D" command). DUP duplicates lines (LCA "" command). LADD adds blank lines (LCA "A" command). INSERTI inserts line with specified text (LCA "A" command followed by overtype). MERGE merges lines from the stack (LCA "J" or "G" commands). LOWERCAS translates to lower case (LCA "W" command). UPPERCAS translates to upper case (LCA "U" command).

There are a number of editing commands which perform a function for which there is no LCA equivalent. BLANK and KEEP blank out specified columns. CHANGE replaces the characters which match a text search value with specified characters. FORMAT condenses the words in a series of lines into fewer lines by using the maximum number of words that will fit in a column range. GETx inserts the text lines of another entity (member, POWER job entry, JES data sets, message, etc.). OVERLAY places specified text in specified columns. PROPAGAT copies text from one column range to another. RESEQ puts sequence numbers in specified columns. SEPARATE spreads the words in a series of lines over more lines to make them easier to edit. SORT reorders lines based on the values in specified column ranges. SPLIT cuts a line into two pieces.

The editing commands which shift data in columns are sometimes preferable to their LCA equivalents because you can specify the column range they operate upon rather than use the session range. CENTER centers text in a specified column range. JUSTIFYL and JUSTIFYR move text as far as possible to the left or right, respectively, within a column range. SHIFT moves text a specified number of columns left or right within a column range.

Inter-User Mail

BIM-EDIT provides inter-user mail facilities. When another user has sent you a message, a notice will be posted to you that there is mail waiting for you. If you are logged on, the notice will appear on your information line and your terminal alarm will sound the next time you transmit your screen. Otherwise the notice and the audible alarm will occur when you next log on.

You can display your "incoming" message using the OPEN command. When you are finished reading it, you can terminate the MAIL session with END if you wish to save the message in your "set-aside" message queue or DISCARD if you wish to delete it. Messages are assigned unique ID numbers by BIM-EDIT -- you can display a specific message from the set-aside queue with OPEN by specifying its ID number.

With the AUTOMAIL operand of the DEFINEU and ALTERU commands, you can choose to have some types of mail 'automatically opened' and displayed in the line-2 message area.

To send a multi-line message to another user, make a member and use the MAIL command to send a copy of its text. Use MAILI to send a single specified line of text to another user. You can display a list of the users on your system using the LIBRARYU command. The messages you send will reside in your "outgoing" message queue until they are opened by the recipient.

The LIBRARYQ command displays a list of all messages in your "incoming", "set-aside", and "outgoing" message queues. The PURGEQ command can be used to delete any of these messages.

Two other commands that are relevant to mail processing are GETQ, which inserts a copy of all incoming messages in your current session, and ATTACHX, which allows a user to process mail for another user, if so authorized.

Each user has a special area known as the mail log (\$MAIL). \$MAIL is a complete trace of your recent outgoing and incoming messages, regardless of whether they have been deleted. \$MAIL can be viewed at any time by entering:

```
=> list $mail
```

When your \$MAIL reaches the maximum number of lines specified by your System Administrator, the oldest lines will be reused for new lines.

Access to Batch Processing

The most common use for BIM-EDIT is to collect and modify information which will ultimately be processed in batch mode to produce printed output or some other form of calculational results.

Processing in batch mode operates by "queuing". When you present the contents of a member to be processed, it becomes a "job" moving through the batch processing system in automatic stages, and you may use BIM-EDIT for other work while it does so. The job may wait in a queue until other jobs complete and free up resources it needs to process. Then it will process and may produce output that may wait in a queue until the resources are available to print it.

On VSE, batch processing is controlled by a program called POWER, the input queue is composed of the RDR job entries and the output queue is composed of LST and PUN job entries.

On MVS, batch processing is controlled by JES, the input queue is composed of input data sets and the output queue is composed of sysout data sets.

Presenting Members for Batch Processing

Four commands are available to present members for batch processing. The SUBMIT command presents a member to be processed "as is" (the member must contain "Job Control Language"). COMPILE performs like SUBMIT except it supplies Job Control Language for a program compilation based on the TYPE attribute. PROCESS presents a member to be processed in a manner indicated by its TYPE attribute (PROCESS will either perform a SUBMIT or a COMPILE). PRINT transfers the text of the member directly for output.

A member presented by SUBMIT, COMPILE or PROCESS can contain INCLUDE commands to incorporate the contents of other members as it is presented.

Accessing and Modifying Processing Results

The DA command can be used to display status while processing is going on.

Commands are provided to access the results of batch processing. LIBRARYP displays a list of the entries which are currently accessible to you. LISTP displays the text of an entry. GETP inserts the text of an entry into the current session.

Commands are also provided to modify POWER job entries and JES jobs and data sets. ALTERP changes the attributes of an entry. FALTERP displays and changes the attributes of an entry. HOLD makes an entry unavailable for processing. RELEASE makes an entry available for processing. CANCEL (on MVS only) terminates an entry. PURGEP deletes entries. DISCARD ends the current LISTP session and deletes the entry.

Although the command names are the same under VSE and MVS, the operands and the way they function are somewhat different. Take care to reference the command description for whichever version of BIM-EDIT you are using.

Access to POWER job entries and JES jobs and data sets is restricted as described under "Security" later in this chapter.

Job Completion Notification (VSE only)

BIM-EDIT contains two facilities that may be used to have a user notified when a batch job completes.

The first facility utilizes the same POWER interface that ICCF uses. This facility is activated in BIM-EDIT by setting system variable MMPNOTFY to "1" and cycling the BIM-EDIT system. When this facility is active, POWER will generate job completion messages for any submitted jobs containing the NTFY=YES or NTFY=(,userid) operands on the * \$\$ JOB statement. These messages will be sent to the BIM-EDIT user as MAIL entries. Since this facility is the same one used by ICCF, you can NOT use this facility if you also run ICCF in the same VSE machine. You can also only activate this facility in one BIM-EDIT system per VSE machine. Once you activate this facility, ICCF will fail to startup until you de-activate this facility by setting variable MMPNOTFY to "0" and cycling BIM-EDIT.

The second job notification facility in BIM-EDIT uses a new POWER GCM feature that was added in VSE/ESA 2.1. This facility is automatically activated at BIM-EDIT startup if your system is VSE/ESA 2.1 or above. This facility uses the NTFY operands of the DEFINEU, ALTERU, and SUBMIT to determine whether completion messages should be generated. The DEFINEU and ALTERU commands provide the default setting for the NTFY operand of the SUBMIT command. If you have your own procedures for submitting jobs, you can use the system variable SIBNTFY to control this facility.

If you are running VSE/ESA 2.1 or above, and do not run ICCF, you can use both of these facilities at the same time.

Access to VSE Sublibraries and MVS PDSs

One of the major uses of BIM-EDIT is to manage computer program "source code". Besides the BIM-EDIT library, computer program source code is often stored in areas provided by the operating system. In the case of VSE, this storage is in the system sublibraries. In the case of MVS, this storage is in Partitioned Data Sets (PDSs). BIM-EDIT provides features to directly access members in these areas.

- ATTACHD sets the sublibrary or PDS which will be used to access members if one is not specified.
- CATAL copies a member from BIM-EDIT to a sublibrary or PDS.
- DEFINED creates a member in a sublibrary or PDS.
- EDITD creates an edit session of a member in a sublibrary or PDS; when the session is SAVED, the member will be updated.
- GETD inserts the text of a member from a sublibrary or PDS into the current session.
- LIBDS performs a LIBR SEARCH command and displays the results (VSE only).
- LIBRARYD lists the members in a sublibrary or PDS.
- LISTD displays the text of a member in a sublibrary or PDS.
- PURGED deletes a member in a sublibrary or PDS.
- On MVS, SUBMITD submits a member for batch processing.

Although the command names are the same under VSE and MVS, the operands and the way they function are somewhat different. Take care to reference the command description for the version you are using.

References in a command to a VSE sublibrary member or an MVS PDS members can take one of several forms:

- if only the member name is specified, it refers to the member within the current sublibrary or PDS as set by the ATTACHD command.
- if the member name is omitted, the last referenced member will be used.
- if the sublibrary name or PDS name and the member name are specified, separated by a period (.), the member within the specified sublibrary or PDS will be used.
- on MVS, the PDS name can be followed by the member name in parentheses.

Access to VSE sublibraries or MVS PDSs is restricted as described under "Security" next in this chapter.

Security

It is often detrimental for everyone to have access to everything. BIM-EDIT provides facilities to control access to:

- Members and libraries
- POWER job entries or JES jobs and data sets
- Mail
- VSE sublibraries or MVS PDSs
- Commands

The foundation of security is verifying the identity of the individual who is making a request. This is done on BIM-EDIT, as with most systems, through the use of a password. Do not post your password and do not let others know your password. If you feel your password has been compromised, use the PASSWORD command to change it. When you are using batch utility, use the PEND command to create a temporary password so that your permanent password cannot be read within the batch processing system.

Library Security

For each library in the BIM-EDIT system, your System Administrator has assigned you an access level. Seven levels of access are defined, each level allowing greater access than the previous level:

NULL	Access is completely denied.
EXEC	Members may be executed as procedures.
LIST	Member text may be listed or copied.
EDIT	Member text may be edited.
DEF	Members may be defined, purged, altered.
DEFL	Libraries may be defined, purged, altered.
DEFS	Security entries may be defined, purged, altered.

Normally, only System Administrators have a library access level higher than DEF. The System Administrator normally has DEFS access to all libraries.

The display produced by the LIBRARYL command lists all libraries for which you have EXEC or higher access. Your defined access level is shown for each library.

Member Security

Individual members can be assigned a password to protect them from unauthorized access. Passwords are defined using the PSWD operand of the DEFINE command for new members, or the ALTER command for existing members.

Passwords can be removed from members or changed using the PSWD and NEWPSWD operands of the ALTER command.

Access to password protected members requires that the PSWD operand be specified on each command that edits, lists, renames, submits, includes, purges, or otherwise accesses the member.

POWER Job Entry and JES Data Set Security

Access to POWER job entries or JES data sets is determined by the exit routine BIXPWQA. Normally, this routine only allows you to view, purge, or alter, entries that are assigned to your user ID.

Mail Security

You have access only to messages sent to you and those sent by you but not yet opened by their recipients. If your System Administrator has authorized it, you may have the same access for selected other users.

VSE Sublibrary or MVS PDS Security

Access to members in VSE sublibraries or MVS PDSs (depending on which operating system you run BIM-EDIT under) is "piggybacked" on BIM-EDIT library security. When an access is made, it is handled as if BIM-EDIT had a library of the same name as the VSE sublibrary or the MVS PDS. Based on the type of access requested, it will be accepted or rejected according to the access level you have been assigned for the hypothetical library. For example, if you attempt to PURGED a member in LIBRARY.U728 and your access level for that library is EDIT, the command will be rejected.

Command Security

Access to some commands is restricted to users with administrator or operator command access level. The commands so restricted are those related to setting library security and those related to operations phenomena such as backing up and restoring the BIM-EDIT libraries.

Some other features may be restricted at your site, such as the use of the STAMPCL and AUDITCL commands. If your site has modified or added commands, they may be subject to special security. (SUBMIT is sometimes so modified.)

For a complete discussion of security, see the "When Valid" section of the detailed description of the command and the BIM-EDIT System Reference Manual.

(If you are in a hurry to learn the basics of BIM-EDIT, you may at this point skip forward to Chapter 4, Commands, since the remaining sections in this chapter provide overviews of somewhat advanced topics. However, most users will find this background information useful at some point.)

Member Change Controls

Three facilities are provided to allow you to control changes to BIM-EDIT members. Use these for members which have sensitive implications, for example:

- The tables which control security
- Production program source
- Data changed by many people

The change control facilities, from the least to the most control, are:

Stamping

When a member has the attribute STAMP set to ON, each line will record a one letter code for the type of change and the date of the last change to that line. You can display these stamps in the LCA area using the SCREEN STAMP command. The STAMPCL command can be used to clear existing stamps (the use of STAMPCL can be restricted).

Stamping does not use a lot of resources or storage but it does not record exactly what was changed or provide any indication of deleted lines.

Auditing

When a member has the attribute AUDIT set to ON, a complete line-by-line trace of all changes to the member is kept. You can display the audit trail in the text display area by using the SCREEN AUD command. You can use the AUDITI and AUDITF commands to write comments to the audit trail. You can delete existing audit trail entries using the AUDITCL command or by altering the AUDIT attribute to OFF (use of these commands can be restricted).

Auditing records exactly what was changed including deleted lines but the audit trail will consume a noticeable amount of disk space and processing time.

When an audit trail exists, the AUDITRL command is available to "undo" the effect of edits, using the "before" lines in the audit trail. The AUDITSM command can be used to produce a summary of the net changes represented by an audit trail.

Checkin/Checkout

When a member has attribute CHECK set to ON, it cannot be edited directly. Instead, a working copy for editing must be created using the CHECKOUT command. The only way the working copy can update the master member is through the CHECKIN command, which can be restricted by library security. A CHECKPUR command is provided to delete a working copy without checking it in.

Of course, these three facilities may be used together, the most likely combination being auditing and checkin/checkout.

For more information on change control features, see Chapter 6 of this manual.

(One other feature which can be thought of in this context is the purge control feature, whereby a copy of all purged members are kept in an administrative library. This feature is discussed in the BIM-EDIT System Reference Manual.)

Procedures

BIM-EDIT provides the ability to execute series of commands as if they were one command. Quite elaborate procedures can be set up. Facilities are provided to execute commands over-and-over, to execute commands only if certain conditions exist, and to interact with the terminal. Some of the standard BIM-EDIT commands are in fact procedures.

Procedures are simply members containing commands. They can be invoked exactly like a command if they are in the current library and their name is shorter than eight characters or if they are in a command table. Otherwise the EXECUTE command is used to invoke them.

BIM-EDIT also supports REXX procedures. This support is described in the BIM-EDIT System Reference Manual, in "Chapter 2. Writing Procedures".

Procedures depend on "variables", which are used to store numeric or text values, and on some procedure-specific commands.

Variables

Variables are named areas in which you can temporarily store a number or a string of characters. Variables are created in a procedure using the DECLARE command. Values can be assigned to variables either by using the PARSE command to obtain values from parameters used when the procedure was invoked, or by using the SET, SETD, or SETL commands to obtain values from a constant or another variable. The ADD and SUBTRACT commands can be used to adjust the values of numerical variables. The SHOW command may be used to display the values of variables. Variables disappear when you exit the procedure they were created in.

Variables are used by placing their name, prefixed by an ampersand (&) in a command or in a text line. The value of the variable will replace the name before the command is processed. For example, if a variable MBR contains the value "MYMEMBER", the command

```
EDIT &MBR
```

would have exactly the same effect as

```
EDIT MYMEMBER
```

in a procedure.

BIM-EDIT provides a number of "pre-defined" variables which can be used to access or change BIM-EDIT control values. Using these, a procedure can determine the date, the user ID, or the member it is currently processing. Pre-defined variables are the subject of Chapter 4, Pre-defined Variables, in the System Reference Manual.

Procedure-Specific Commands

Commands are available within procedures to control the order in which other commands are executed. EXIT terminates procedure execution. GOTO transfers execution control to a location elsewhere in a procedure marked by a LABEL command. The IF command executes the command on the line following it only if a specified test is met. EXAMINE allows testing the contents of a session line.

The MAPF command is used in a procedure for interaction with the terminal. It displays a formatted screen and accepts input from the terminal in the form of modified variables.

A series of commands are provided to transfer a block of lines following them to a BIM-EDIT destination: AUDITF transfers to the member audit trail, CONSOLEF to the operating system console, DISPLAYF to a display session, INSERTF to the member, LOGF to \$LOG, PRINTF to the printer, PUNCHF to the punch, REPROF to tape, STACKF to \$STACK, and SUBMITF to batch processing. OUTPUTF transfers following lines to any of these destinations based on an operand.

READ and BROWSE allow a procedure to process many members from a library. READL and BROWSEL allow a procedure to process many libraries. READP allows a procedure to obtain information about POWER job entries or JES data sets.

The SNAP command is used to provide more information when a procedure terminates abnormally.

Each command description contains a "Use in a Procedure" section which discusses how that command will operate in a procedure. It describes the codes that a procedure can test to determine the success of a command and any secondary effects of the command. If you are not involved in writing procedures, you may safely ignore this section of the command descriptions.

For a complete discussion of procedures, see the BIM-EDIT System Reference Manual.

Batch Utility

BIM-EDIT can be executed in a batch processing mode. Most commands are available in batch although some facilities that are screen-oriented (such as the LCA commands) are not. Some features are only available in batch processing mode. A version of BIM-EDIT batch is provided which can communicate across an SNA LU6.2 link to BIM-EDIT running in another machine. BIM-EDIT batch can also be run from the operating system console, which is useful under some specialized situations.

In batch processing mode, your input commands and any messages they generate will be printed, as will LIBRARY results and similar information requests. However, session output will not be printed. Any online user is eligible to run batch processing mode (even at the same time as the user is using BIM-EDIT online). A user running in batch cannot make permanent changes in the user environment - \$LOG, \$STACK, any open sessions, and changes in the user control variables are not saved when a user running in batch logs off.

Batch utility commands are provided to copy the BIM-EDIT library to and from back-up tapes. BACKUPP copies the entire BIM-EDIT library to tape in a "physical" format, meaning that individual members cannot be copied back. It is used rarely. BACKUPG copies the entire BIM-EDIT library to tape in a "logical" format, meaning that individual members can be copied back (by the RESTORE command). BACKUPS copies selected BIM-EDIT entities to a logical backup tape for archival purposes or to transfer between BIM-EDIT sites. All sites should copy the BIM-EDIT libraries to tape every night using BACKUPG and save the tapes for at least a week.

For bringing members from other editing systems into BIM-EDIT, the batch LOADx commands are provided. Members are imported by direct access to the other system's library.

The contents of a tape can be read directly in batch with the INCLUDE command.

The batch utility can be used for printing the text of a member or the results from executing a procedure. The HEADF command stores the lines which follow it to use as the page heading. EJECT skips to the top of a page. SEGMENT starts a new print data set (useful when batch utility is run from the console in VSE). There are also predefined variables which can be used to control print related phenomena such as the length of pages, the handling of page numbers, and the printing of input commands and responses.

The batch utility can also export member data out of BIM-EDIT. REPRO copies the text of a specified member to tape. REPROL copies the text of all members in a library. REPROI writes a specified line to tape (it is used to intersperse control commands with the text). The PUNCH, PUNCHL, and PUNCHI commands perform the same functions but write the output to the punch device. AUDITRP copies the audit trail of a specified member to tape.

For a complete discussion of batch utility, see the BIM-EDIT System Reference Manual.

Applications Interface

BIM-EDIT can be executed as if it were a subroutine of another program. In essence, a program can present commands to BIM-EDIT to be processed and obtain responses back from BIM-EDIT. The restrictions on which commands can be executed in this mode and the results that will be generated are almost identical to those for Batch Utility execution.

Four additional commands are provided for communication with an application program: SEND, SENDF, SENDI, and SENDN.

For a complete discussion of the applications interface, see the BIM-EDIT System Reference Manual.

Customization

BIM-EDIT can be modified to fit a particular site's needs.

Command Customization

Many sites customize BIM-EDIT by altering the function of supplied commands or adding additional commands. This can be done in several ways:

- Modify a supplied procedure. Several BIM-EDIT commands are provided as procedures. SUBMIT and COMPILE are commonly modified.
- Write a new procedure and place in the user or site tables as a command.
- Provide access to CICS programs directly from BIM-EDIT. A CICS program that operates from data on the transaction line can be set up so that BIM-EDIT will invoke it when the transaction code is entered and it will return to BIM-EDIT when complete.
- Write a procedure to front-end a command. You can write a procedure to supplant a command by the same name. You can then check or modify operands before executing the command or perform other actions such as specialized logging. The BYPASS and AUTHORIZ commands are provided for procedures of this type.
- Write a new command or modify a supplied command. BIM-EDIT provides the ability to create your own programs that will function exactly as a BIM-EDIT command. Under some circumstances, BIM will supply you with the assembly language source as a starting point for your own modifications.

If a command does not operate as it is described in this manual, it may be due to customization done by your site. When a command is often customized, it is noted in the manuals.

Other Customization

Many sites modify the member type characteristics used by the DEFINE command.

The documentation source is supplied with BIM-EDIT and can be modified to produce new HELP text or new printed manuals.

BIM-EDIT provides "hooks" for some exit routines. These are primarily related to interfacing with the external computer system and pertain to functions like LOGON, SUBMIT, and POWER job entry or JES data set access.

Some control tables are available to be modified, in addition to the command tables.

Obviously, customization needs to be weighed carefully, since it costs effort to do, may confuse support and documentation, and may need to be revised for each new release of BIM-EDIT. But in many environments, there will be some customizations that have good justification.

For a complete discussion of customization, see the BIM-EDIT System Reference Manual.

Chapter 4. Commands

This chapter describes the most commonly used BIM-EDIT command-line commands (as distinguished from the LCA commands described in Chapter 5). Other command-line commands, including those useful only for batch processing or in building procedures, or those which are normally only needed for managing BIM-EDIT, are described in the BIM-EDIT System Reference Manual.

The commands in this chapter are presented in alphabetical order for easy reference. Each command description provides:

- The functional description of the command.
- Table of all operands allowed, in the order they must appear when used without keyword names. The keyword name, whether the operand is required or optional, and the meaning of the operand value are described.

For example, an operand table like this one for the ALTERL command

Required Operands	
LIB	is the library to be altered.

Optional Operands	
TITLE	is a title of, comment about, or description of the library.
USER	is purely documentary. When the library was created with the DEFINEL command, if USER was not specified, USER was set to the logon user ID.

indicates the following would be acceptable:

ALTERL ED51A,"LIB TITLE",MWD Sets library ED51A title to "LIB TITLE" and user to MWD.

ALTERL USER=ABC,LIB=ED51A Sets library ED51A user to ABC

and the following would be unacceptable:

ALTERL TITLE="LIB TITLE" Required operand LIB missing

ALTERL TITLE="LIB TITLE",ED51A Positional operand follows keyword operand

- "When Valid". Describes any limits on command usage due to the status of BIM-EDIT or due to the command, library, or POWER or JES authority of the user. If this subheading is omitted, the command is always valid.
- "Use in a Procedure". Typically indicates "return codes" generated by the command when executed in a procedure.
- Examples

User Command List

The following tables list all of the BIM-EDIT online user commands. The tables are separated into useful categories. The shortest abbreviation for the command is also shown.

More information about each command may be found in the pages which follow alphabetically by the command name.

Commands Used to Manipulate Members		
ALTER	ALT	Alter member attributes.
ATTACH	ATT	Set library default for member access.
AUDITCL	AUDCL	Delete member's audit trail to specified date.
AUDITI	AUDI	Write specified comment line to audit trail.
AUDITRL	AUDRL	Back out member edits using audit trail.
AUDITSM	AUDSM	Summarize member's audit trail.
CATAL	CAT	Copy a member to a VSE sublibrary or MVS PDS.
CHECKASN	CHKA	Move the working copy of a member to another user.
CHECKIN	CHKI	Replace a master member with its working copy.
CHECKOUT	CHKO	Create the working copy of a master member.
CHECKPUR		Delete the working copy of a master member.
COMPILE	COMP	Submit a program compilation for batch processing.
COPY		Create copy of an existing member.
DEFINE	DEF	Create new member.
DISPLAY	DI	Display text of member, \$LOG, \$MAIL, or \$STACK.
EDIT	ED	Create edit session of an existing member.
FALTER	FALT	Display / alter member attributes - formatted screen.
FCOPY		Create copy of an existing member - formatted screen.
FDEFINE	FDEF	Create a new member - formatted screen.
FRENAME	FREN	Alter the name of a member - formatted screen.
GET		Alter session text - insert text of member.
INQUIRE	INQ	Display member attributes.
LIBRARY	LIB	Display list of members in a library.
LIST	LI	Display text of member, \$LOG, \$MAIL, or \$STACK.
MODIFY	MOD	(See ALTER).
MOVE		Move or rename a member.
PROCESS	PROC	Process (SUBMIT, COMPILE or EXECUTE) a member.
PURGE	PUR	Delete member.
RENAME	REN	Alter the name of a member.
ROLLBACK		(See AUDITRL).
SCAN		Search multiple members for a specified pattern.
STAMPCL		Blank out member's date stamps.
SUBMIT	SUB	Submit a member for batch processing.

Commands Used to Manipulate Libraries		
ALTERL	ALTL	Alter library attributes.
ATTACH	ATT	Set library default for member access.
DEFINEL	DEFL	Create new library.
FALTERL	FALTL	Display / alter library attributes - formatted screen.
FDEFINEL	FDEFL	Create new library - formatted screen.
LIBRARY	LIB	Display list of members in a library.
LIBRARYL	LIBL	Display list of libraries you can access.
MODIFYL	MODL	(See ALTERL).
PURGE	PURL	Delete empty library.
SCAN		Search multiple members for a specified pattern.

Commands Used to Alter Session Text		
BLANK	BL	Blank specified columns.
CENTER	CEN	Center in specified columns.
CHANGE	C	Replace specified pattern with a string.
DELETE	DEL	Delete lines.
DUP	DUP	Duplicate current line.
FORMAT	FORM	Wordwrap lines into paragraph.
GET		Insert text of member.
GETD		Insert text of VSE sublibrary or MVS PDS member.
GETP		Insert text of POWER job entry or JES data sets.
GETQ		Insert text of all incoming messages.
INSERTI	INS	Insert specified line after current.
JUSTIFYL	JUSTL	Left justify lines in a column range.
JUSTIFYR	JUSTR	Right justify lines in a column range.
KEEP		Blank all but a column range.
LADD	LA	Insert blank lines after current.
LOWERCAS	LOW	Translate lines to lower case.
MERGE		Overlay non-blank \$STACK text.
OVERLAY	O	Put specified string at columns.
PROPAGAT	PROP	Copy from columns to others.
RESEQ	RES	Put sequence numbers at columns.
SEPARATE	SEP	Break paragraph into lines.
SEQCHECK	SEQCK	Check the order of the current session lines.
SHIFT	SH	Move right or left in columns.
SORT		Reorder lines based on columns.
SPLIT	SP	Break line at column or string.
SQUEEZE	SQU	Remove embedded blanks.
STACK	ST	Copy lines from current session to \$STACK.
STACKI		Write a single line to \$STACK.
UPPERCAS	UPP	Translate lines to upper case.

Commands Used to Change Position in a Session		
BACK	BA	Position session display up one or more screens.
BOTTOM	B	Position session display at down-most screen.
CURSOR	CUR	Set cursor to column set by last search command.
DOWN	N	Position session display down by one or more lines.
FIND	F	Search session downward to pattern at a column.
FINDUP	FU	Like FIND, except search upward.
FORWARD	FO	Position session display down one or more screens.
LOCATE	L	Search session downward to specified pattern.
LOCATEU	LU	Like LOCATE, except search upward.
NEXT	N	Position session display down one or more lines.
NFIND	NF	Search down to line WITHOUT specified pattern at column.
NFINDUP	NFU	Like NFIND, except search upward.
POSITION	P	Position session to a specified line number.
SEARCH	SRCH	Like LOCATE, except from top of session.
SWAP	SW	Position session display to alternate marker.
TOP	T	Position session display to up-most screen.
UP		Position session display up one or more lines.
VIEW		Position session display left or right.

Commands Used to Access POWER Job Entries		
ALTERP	AP	Alter POWER job entry attributes.
COMPILE	COMP	Submit a program compilation for batch processing.
DQ		(See LIBRARYP).
FALTERP	FALTP	Display / alter POWER job entry attributes.
GETP		Alter session text - insert text of POWER job entry.
HOLD		Put a POWER job entry on hold.
INQUIREP	INQP	Display list of currently executing job output.
LIBRARYP	DQ	Display list of POWER job entries.
LISTP	LP	Display text of POWER job entry.
MODIFYP	MODP	(See ALTERP).
PRINT		Print a member.
PROCESS	PROC	Process (SUBMIT, COMPILE or EXECUTE) a member.
PURGEP	PP	Delete POWER job entry.
RELEASE	REL	Take a POWER job entry off hold.
RELIST		(See PRINT).
SUBMIT	SUB	Submit a member for batch processing.

Commands Used to Access JES Jobs or Data sets		
ALTERP	AP	Alter JES job or data sets attributes.
CANCEL	CANC	Terminate a JES job.
COMPILE	COMP	Submit a program compilation for batch processing.
DQ		(See LIBRARYP).
FALTERP	FALTP	Display / alter JES job or data sets attributes.
GETP		Alter session text - insert text of JES data sets.
HOLD		Put JES job or data sets on hold.
INQUIREP	INQP	Display list of JES data sets for a job.
LIBRARYP	DQ	Display list of JES jobs or data sets you can access.
LISTP	LP	Display text of JES data sets.
MODIFYP	MODP	(See ALTERP).
PRINT		Print a member.
PROCESS	PROC	Process (SUBMIT, COMPILE or EXECUTE) a member.
PURGEP	PP	Delete JES job or data sets.
RELEASE	REL	Take JES job or data sets off hold.
SUBMIT	SUB	Submit a member for batch processing.
SUBMITD	SUBD	Submit a PDS member for batch processing.

Commands Used to Process Inter-User Mail		
ATTACHX	ATTX	Set user for mail access (act as proxy for).
GETQ		Alter session text - insert text of all incoming messages
LIBRARYQ	LIBQ	Display list of your mail messages.
LIBRARYU	LIBU	Display list of users.
MAIL		Send message - member - to another user.
MAILI		Send message - single line - to another user.
MAILSESS		Send message - current session - to another user.
OPEN		Display text of mail message.
PURGEQ	PURQ	Delete mail message.

Commands Used to Access VSE Sublibrary Members		
ATTACHD	ATTD	Set VSE sublibrary default for access.
CATAL	CAT	Copy a member to a VSE sublibrary.
COPYD		Create a copy of a VSE sublibrary member.
DEFINED	DEFD	Create new VSE sublibrary member.
DISPLAYD	DID	Display text of VSE sublibrary member.
EDITD	EDD	Create edit session of VSE sublibrary member.
GETD		Alter session text - insert text of VSE sublibrary member
INQUIRED	INQD	Display attributes of a VSE sublibrary member
LIBDS		Display the output of a VSE LIBR SEARCH command.
LIBRARYD	LIBD	Display list of VSE sublibrary members.
LISTD	LID	Display text of VSE sublibrary member.
PURGED	PURD	Delete VSE sublibrary member.
RENAMED	REND	Alter the name of a VSE sublibrary member.

Commands Used to Access MVS Partitioned Data Set Members		
ATTACHD	ATTD	Set PDS default for access.
CATAL	CAT	Copy a member to a PDS.
COPYD		Create a copy of a PDS member.
DEFINED	DEFD	Create new PDS member.
DISPLAYD	DID	Display text of PDS member.
EDITD	EDD	Create edit session of PDS member.
GETD		Alter session text - insert text of PDS member.
LIBRARYD	LIBD	Display list of PDS members.
LISTD		Display text of PDS member.
PURGED	PURD	Delete PDS member.
RENAMED	REND	Alter the name of a PDS member.
SUBMITD	SUBD	Submit a PDS member for batch processing.

Miscellaneous Online Commands		
CONSOLEI	CONSI	Write specified line to the operating system console.
DA		Display operating system's processing status.
DISCARD	DISC	End current session; delete associated entry or message.
DISPLAYS	DS	Display a list of all active sessions.
END		End current session.
EXECUTE	EX	Process text of a member as commands.
FSESSION	FSESS	Alter session attributes - formatted screen.
HELP		Display information about a command.
INCLUDE	INCL	Incorporate lines from a member during processing.
IND\$FILE		PC File Transfer Interface.
KEYS	KEY	Display / alter function key settings - formatted screen.
LIBSDL		Display output of VSE LIBR LISTDIR SDL command.
LOGI		Write specified line to \$LOG.
LOGOFF	LOGOF	Exit from BIM-EDIT.
PASSWORD	PAS	Alter user's own password.
QUALIFY	QUAL	Display all lines of a session having specified pattern.
QUIT	Q	End current session; retain any edits.
REFRESH	REF	Recreate a display session.
RESET		Forget pending LCA bracket command.
ROTATE	ROT	Select session to display.
SAVE	Q	End current session; retain any edits.
SCREEN	SCR	Set session display modes.
SESS	SS	Alter session attributes.
SET		Set PF key commands or BIM-EDIT controls.
SHOW		Display BIM-EDIT general or PF key information.
VTOC		Create DASD VTOC display session.
&		Redisplay commands after processing.
=		Process last command again.
?		Display last command processed.
??		Display last command stack.

<		Select other logical screen.
---	--	------------------------------

ALTER

Use ALTER to alter the attributes of an existing BIM-EDIT member.

ALTER may also be entered as ALT, MODIFY, or MOD.

Optional Operands	
MEM	is the member to be altered. If MEM is not specified, the "last referenced member" is used.
TYPE	specifies a new member type. For program source members, TYPE is typically used by the COMPILE command to determine which compiler to create Job Control Language for. TYPE can be altered to anything but is typically one of the selections described in the DEFINE command. Altering TYPE does not change the contents or other attributes of the member.
TITLE	specifies a new comment about or description of the member. TITLE can be up to 40 characters in length.
ATTR	is an attribute with site-defined usage.
AUDIT	Specify "ON" to start member auditing. Specify "OFF" to stop member auditing and clear any current audit trail. You cannot alter AUDIT while a member is in a checkout relationship. If your System Administrator has set predefined variable MMPAUCTL to "1", only the System Administrator can ALTER AUDIT=OFF.
CASE	Specify "U" if updated text lines are to be translated to upper case. Specify "M" if no translation is to occur. Only new and updated lines are affected by CASE. Compare UPPERCAS command.
CHECK	specifies whether member is under checkout\checkin control. Specify "ON" if editing can be done only on a checked out copy. Specify "OFF" if member can be edited directly. You cannot alter CHECK while a member is part of a checkout relationship.
LIBDEF	is an attribute with site-defined usage.
NULLS	specifies whether the trailing blanks on a line are replaced with nulls when displaying the member.
SEQ	specifies a new column range where the RESEQ command places sequence numbers.
TCOL	specifies a new list of columns used for tabbing. Up to 12 tab settings may be specified. They are entered in the format "cc-cc-cc".

Optional Operands (continued)	
USER	is purely documentary. It can be set to any value -- typically the value is used to determine ownership when culling junk members from the library. It can be up to 8 characters.
ZONE	specifies a new column range in which edit and search commands process.
FCOL	specifies a new column used by the FIND, FINDUP, NFIND, and NFINDUP commands.
STAMP	Specify "ON" if member stamping is desired. "OFF" turns member stamping off. You cannot alter STAMP while a member is part of a checkout relationship. If your System Administrator has set predefined variable MMPSPCTL to "1", only the System Administrator can ALTER STAMP=OFF.
WRAP	Specifies whether BIM-EDIT/XP Edit sessions of this member are to use word-wrap mode.
PSWD	Specify a member password if the member is password protected. The password can be up to 8 characters in length consisting of any character. If the member is not password protected this field is ignored.
NEWPSWD	Specify an optional member password if the member password is being changed or the member is being password protected for the first time. The password can be up to 8 characters in length consisting of any character. If the member password is being removed this field must be specified as "NO".
PGCTL	<p>is used to specify specific PURGE options for this member. The PGCTL has four values, "NORMAL", "ALWAYS", "NEVER", or "LOCKED".</p> <p>"NORMAL" specifies that the member will use the active system option for purge control. (See MMPPGCTL).</p> <p>"ALWAYS" specifies that the member will always be written to the \$SIT.PURGE library regardless of the MMPPGCTL setting.</p> <p>"NEVER" specifies that the member will never be written to the \$SIT.PURGE library when purged.</p> <p>"LOCKED" specifies that the member cannot be purged.</p> <p>The default is "NORMAL".</p>
LOCK	is used to lock a member so that it can no longer be updated. LOCK may be specified as "NO" or "OFF", or "YES" or "ON". "NO" or "OFF" allows the member to be updated. "YES" or "ON" causes the member to be locked so it can no longer be updated without first setting the LOCK off.

For a more detailed discussion of member attributes, see the DEFINE command.

The FALTER command may provide a more convenient way to alter a member's attributes. FALTER displays all member attributes and allows you to change an attribute simply by overtyping its current value.

When an EDIT or LIST session of a member is created, the member's attributes become the session's attributes. Altering a member's attributes after an EDIT or LIST session of the member is created has no effect on that session. If you want to affect the attributes of an existing session, use the SESS command.

If you ALTER the attributes of a slave member in a checkout relationship, the changes are lost when the CHECKIN is done.

ALTER sets the "last referenced member".

When Valid

The user must have DEF level access for the MEM library.

Use in a Procedure

Return Codes:

OK	Successful.
CK	Member is part of a checkout relationship.
NF	Member not found.
SC	Inadequate access level.

ALTER sets the TXM variables to the attributes of MEM. See Chapter 4, Predefined Variables, in the BIM-EDIT System Reference Manual or HELP TXM.

Examples

Alter title for member ESAP030:

```
=> alter esap030,title='online vendor activity'
```

For member RMAP050 in library 3982, alter the type and case:

```
=> alt 3982.rmap050,type=cobol,case=u
```

Set tabs for member SYREXIO:

```
=> alter syrexio,tcol=8-16-20-24-40-78
```

ALTERL

Use ALTERL to alter the attributes of a library.

ALTERL may also be entered as ALTL, MODIFYL, or MODL.

Required Operands	
LIB	is the library to be altered.

Optional Operands	
TITLE	specifies a new comment about, or description of the library.
USER	is purely documentary. When the library was created with the DEFINEL command, if USER was not specified, USER was set to the logon user ID.

(The FALTERL command may provide a more convenient way to alter a library's attributes. FALTERL displays all library attributes and allows you to change an attribute simply by overtyping its current value.)

When Valid

The user must have DEFL access level for the library prefix of the LIB library (the part of the library name preceding an imbedded period). If LIB has no prefix, the user must have system-wide DEFL access.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Library not found.
SC	Inadequate access level.

ALTERL sets the TXL variables to the attributes of LIB. See Chapter 4, Predefined Variables, in the BIM-EDIT System Reference Manual or HELP TXL.

Examples

Alter title for library 4216:

```
=> alterl 4216,'user #4216 library'
```

For library OM20, alter the user:

```
=> altl OM20,user=$SYS
```

ALTERP (VSE version)

Use ALTERP to alter the attributes of a POWER job entry.

ALTERP may also be entered as ALTP, AP, MODIFYP, or MODP.

Optional Operands	
PRM1-4	<p>select the POWER job entry to be altered. The four operands allow you to specify the queue (RDR,LST,PUN,XMT), job name, job number, and segment number. They can be specified in any order, except that segment, if specified, must follow job number. Which entry will be selected depends on the operands specified:</p> <p>queue, job name, job number, segment</p> <p>The specified queue will be searched for the entry for the specified name, number, and segment.</p> <p>queue, job name</p> <p>The specified queue will be searched for the first entry for the specified name.</p> <p>queue, job number</p> <p>The specified queue will be searched for the first entry for the specified number, without regard to the job name.</p> <p>job number</p> <p>job name</p> <p>The LST queue will be searched for the first entry for the specified number or name.</p> <p>queue</p> <p>The specified queue will be searched for the first entry for the job name and number of the "last referenced POWER job entry"</p> <p>If PRM1-4 is not specified at all, the "last referenced POWER job entry" will be used.</p>
CLASS	<p>specifies a new POWER class. POWER uses class to select the resources to use to process an entry. Values are site-specific.</p>
DISP	<p>specifies a new POWER disposition. POWER uses disposition to determine the circumstances of processing an entry. Values are D = dispatchable, K = keep, L = leave, H = hold.</p>
PRI	<p>specifies a new POWER priority. POWER uses priority to determine the order of processing entries.</p>
COPY	<p>specifies a new POWER copy count. POWER uses the copy count to determine how many copies of an entry to process.</p>

Optional Operands (continued)	
DEST	specifies a new POWER destination. POWER uses destination to route output in a networked environment.
SYSID	specifies a new POWER system ID. POWER uses system ID to route output in a multi-CPU environment.
USER	specifies a new POWER user ID. POWER records user IDs in job accounting.
FNO	specifies a new POWER forms ID. POWER uses forms ID to manage output devices.
UINF	specifies a new POWER user information value.

(The FALTERP command may provide a more convenient way to alter a POWER job entry's attributes. FALTERP displays all job entry attributes and allows you to change an attribute simply by overtyping its current value.)

ALTERP sets the "last referenced POWER job entry".

When Valid

Check with your System Administrator to determine the restrictions on access to POWER job entries at your site. The exit routine distributed with BIM-EDIT provides several restriction schemes your System Administrator can select among. (Use of and changes to the BIXPWQA exit routine are described in the BIM-EDIT System Reference Manual.)

Use in a Procedure

Return Codes:

OK	Successful.
NF	Job not found.
PW	Rejected by POWER.
*	Other codes as set by BIXPWQA exit routine.

ALTERP sets the PWR variables to the attributes of the job entry altered.

Examples

Alter a job entry in the LST queue, identifying it with queue and job name:

```
=> alterp lst,asm100,class=a,disp=d
```

Alter a job entry in the RDR queue, identifying it with queue and job number:

```
=> altp rdr,1278,pri=4
```

Alter a job entry in the LST queue, identifying it with just the job number:

```
=> ap 3426,pri=4
```

Alter the last referenced POWER job entry:

```
=> alterp disp=d
```

ALTERP (MVS version)

Use ALTERP to alter the attributes of a JES job or JES data sets.

ALTERP may also be entered as ALTP, AP, MODIFYP, or MODP.

Optional Operands	
QUEUE	specifies the JES queue. QUEUE may be specified as "I" for the input queue, "H" for the held queue or "O" for the output queue.
JOB	selects the JES job to be altered. If JOB contains letters, it specifies a jobname label from the Job Control Language JOB statement. If JOB is entirely digits, it specifies a job number assigned by JES. If not specified, the job from "last referenced JES data sets" will be used.
GROUP	<p>selects the data sets to be altered within the specified job. To alter a JES job (as opposed to data sets), do not specify GROUP.</p> <p>If GROUP is a single letter or digit, it specifies all data sets in the GROUP class in the JES "hold queue". If GROUP is any other multiple character value, it specifies all data sets in the GROUP outgrp in the JES "output queue".</p> <p>An outgrp may be composed of three parts separated by periods in the form "name.qualifier.copy" where "name" is the JES or user-specified name, qualifier is a JES-assigned number, and copy is the copy number. All data sets matching a partial outgrp specification (for example, "name" only) will be altered. To distinguish an outgrp with a one-character name from a class, follow the name with a period.</p> <p>If neither JOB or GROUP is specified, the "last referenced data sets" are altered. (If "last referenced data sets" was specified with a DSET operand, ALTERP will be rejected because JES provides no facility to alter by data set.)</p>
CLASS	specifies a new JES class. JES uses class to select the resources to use to process a job or data sets. Values are site-specific.
PRI	specifies the new JES priority. JES uses priority to determine the order of processing entries. Values are site-specific.
FORMS	specifies the JES print output forms name.
DEST	specifies the JES print destination name.
ODISP	specifies the JES output disposition. Valid specification is HOLD, LEAVE, KEEP, WRITE or PURGE.
FCB	specifies the FCB name to be used for the printer output.
UCS	specifies the universal character set print train identifier.

WTR	specifies the JES output externam writer name.
BURST	specifies the 3800 print burst indicator. Valid values are "yes" or "no".
PRMODE	specifies the JES printer process mode.
FLASH	specifies the output FLASH name.
SAFF	specifies the JES execution system affinity.
XNODE	specifies the JES execution node name.
SRVCL	specifies the JES service class.
SCHED	specifies the JES scheduling environment for the job.
STAT	specifies the desired JES status. STAT may be specified as OPER to place an output queue job on "operator" hold or RELEASE to release a held queue job or an output queue job previously placed on operator hold.

To alter a JES job (as opposed to data sets) specify the JOB operand but do not specify the GROUP operand. To usefully alter a JES job, the job must not have started processing.

ALTERP operates on all data sets within the job that meet the selection.

The relationship between the "output queue" and the "hold queue" is a little confusing. For a data set to be in the "hold queue", HOLD=YES must have been specified or implied by the CLASS on the DD statement that created it and it must never have been RELEASEd or altered to a CLASS which implies HOLD=NO. When a HOLD command is issued for a data set that is in the "output queue", the data set is placed on hold status but remains in the "output queue", and must therefore be accessed by outgrp rather than by class.

(The FALTERP command may provide a more convenient way to alter a job's or data sets' attributes. FALTERP displays attributes and allows you to change them simply by overtyping their current value.)

ALTERP sets the "last referenced JES data sets".

When Valid

Check with your System Administrator to determine the restrictions on access to JES jobs and data sets at your site. The exit routine distributed with BIM-EDIT provides several restriction schemes your System Administrator can select among. (Use of and changes to the BIXPWQA exit routine are described in the System Reference Manual.)

Use in a Procedure

Return Codes:

OK	Successful.
NF	JES job or data sets not found.
*	Other codes as set by BIXPWQA exit routine.

ALTERP sets the JCT, JQE, and PDB variables to the attributes of the job or data sets altered. If GROUP is an outgrp, the JOE variables are also set.

Examples

Change the class of job 1234 to I:

```
=> alterp 1934,class=i
```

Change the class A hold queue output for the last referenced job to class B:

```
=> altp group=a,class=b
```

Change job U782PRT output queue outgrp 2.1 data sets to class A:

```
=> ap u782prt,2.1,class=a
```

ATTACH

Use ATTACH to set the library used by default when accessing BIM-EDIT members (that is, the current library).

ATTACH may also be entered as ATT.

Optional Operands	
LIB	is an existing BIM-EDIT library to use as the default for member access. If LIB is specified as "(null)", you will be detached from your current library, without attaching to a new library. If LIB is not specified, you will be attached to your "home" library.
OPT	Specify "H" or "HOME" to set your "home" library to LIB. Specify "SETH" to only set your "home" library to LIB, this will not change you current attached library.

ATTACH can also be invoked by entering the Y command in the LCA area of a LIBRARYL display.

Detaching from all libraries can be used to test that a procedure does not contain any unwanted dependencies on the current library.

When Valid

The user must have EXEC level access for the LIB library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Library not found.
SC	Inadequate access level.

Your home library is determined by variable SIBHMLB.

Examples

Attach to library "4216":

```
=> attach 4216
```

Detach from any library, in other words, make a "null" attachment:

```
=> att (null)
```

Attach to library "\$SYS.DOC":

```
=> attach $sys.doc
```

ATTACHD (VSE version)

Use ATTACHD to set the VSE sublibrary used by default when accessing VSE source, object, phase, or proc members.

ATTACHD may also be entered as ATTD.

Required Operands	
LIB	is an existing VSE sublibrary you wish to use for the CATAL, DEFINED, DISPLAYD, EDITD, GETD, LIBRARYD, LISTD, LOADD, LOADDL, and PURGED commands. Specify the library name, a period(.), and the sublibrary name. If LIB is not specified, you will be attached to your "home" sublibrary.
OPT	Specify "H" or "HOME" to set your "home" sublibrary to LIB. Specify "SETH" to only set your "home" library to LIB, this will not change you current attached library.

When Valid

BIM-EDIT must be running in its own partition and under VSE/SP release 2.1 or later. The user must have EXEC level access for the LIB library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Library not found.
SC	Inadequate access level.

Your home sublibrary is determined by variable SIBHMLBD.

Example

Attach to VSE sublibrary BIMLIB.ED51M:

```
=> attd bimlib.ed51m
```

ATTACHD (MVS version)

Use ATTACHD to set the Partitioned Data Set used by default when accessing PDS members.

ATTACHD may also be entered as ATTD.

Required Operands	
LIB	is an existing PDS you wish to use for the CATAL, DEFINED, DISPLAYD, EDITD, GETD, LIBRARYD, LISTD, LOADD, LOADDL, and PURGED commands. Specify the full name of the PDS. If LIB is not specified, you will be attached to your "home" PDS.
OPT	Specify "H" or "HOME" to set your "home" PDS to LIB. Specify "SETH" to only set your "home" library to LIB, this will not change you current attached library.

When Valid

The user must have EXEC access level for the LIB library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	PDS not found.
SC	Inadequate access level.

Example

Attach to PDS BIM001.SRCLIB:

```
=> attd bim001.srclib
```

ATTACHX

Use ATTACHX to set the user for processing mail, in other words, to act as a proxy for another user.

ATTACHX may also be entered as ATTX.

Optional Operands	
USER	is the user you wish to process mail for. If USER is not specified, you will revert to processing your own mail.

After you have issued the ATTACHX command, all subsequent electronic mail commands will act upon the USER mail entries. You can LIBRARYQ, MAIL, MAILI, OPEN, and PURGEQ messages on behalf of USER.

However, the MAIL indicator on the information line responds only to your own mail -- not to your proxy user's mail.

ATTACHX is reset by the next logon.

When Valid

The user issuing the ATTACHX must have proxy access to USER. Proxy access is established with the DEFINEX command and revoked with the PURGEX command.

Use in a Procedure

Return Codes:

OK	Successful.
NF	User not found.
SC	Proxy access denied.

Examples

Process mail for user MBX:

```
=> attachx mbx
```

Process your own mail:

```
=> attx
```

AUDITCL

Use AUDITCL to fully or partially clear a member's audit trail.

AUDITCL may also be entered as AUDCL.

Required Operands	
MEM	is the member whose audit trail is to be cleared.
DATE	specifies how much of the audit trail to clear. If DATE is specified as an asterisk (*), the entire audit trail is cleared. Otherwise, DATE is entered in one of the following formats: "mm/dd/yy", "mm/dd/yyyy", "dd/mm/yy" or "dd/mm/yyyy" depending upon the date format set by the System Administrator. In this case, the audit trail is cleared of all entries up to and including DATE.

Optional Operands	
TIME	optionally is used to further qualify how much audit trail to be cleared. If a TIME is entered as hh:mm:ss, the audit trail is cleared of all entries up to and including TIME within DATE.

AUDITCL sets the "last referenced member".

When Valid

If your System Administrator has set MMPAUCTL to a value of "1", AUDITCL is only available to the System Administrator. Otherwise, the user must have DEF access level for the MEM library. MEM must not currently be undergoing editing or be either the master or the slave in a checkout relationship.

Use in a Procedure

Return Codes:

OK	Successful.
CK	Member is part of a checkout relationship.
ED	Member is being edited.
NF	Member not found.
SC	Inadequate access level.

AUDITCL sets the TXM variables to the attributes of MEM.

Examples

Clear the entire audit trail for member APC0100:

```
=> auditcl apc0100,*
```

Clear the audit trail up to Dec 31, 1994 for member XIR2500 (mm/dd/yy format):

```
=> auditcl xir2500,12/31/94
```

Clear the audit trail up to 31 Dec 1994 for member XIR2500 (dd/mm/yy format):

```
=> audcl xir2500,31/12/94
```

AUDITI

Use AUDITI to write a specified line (sometimes called the "immediate" line) as a comment to a member's audit trail.

AUDITI may also be entered as AUDI.

Required Operands	
LINE	is the line to write to the audit trail.

The audit line created is prefixed with the seven characters ".**** ".

Use in a Procedure

AUDITI normally returns OK. (See SIBRETC D.)

Example

Write a comment line to the audit trail:

```
=> auditi 'the following changes resolve problem #39'
```


AUDITRL

Use AUDITRL to back out edits that have taken place against a member that has an audit trail.

AUDITRL may also be entered as AUDRL or ROLLBACK.

AUDITRL has no operands.

To use AUDITRL:

1. Create an EDIT session for the member.
2. After creating the EDIT session, you are looking at the text of the member. Enter the following command so that you will be looking at the audit trail.

```
=> screen aud,on
```

3. After entering the above command, you will be positioned at the top of the audit trail. If you want a complete rollback, that is, if you want to back out all edits that have taken place since auditing was activated, you are positioned correctly. However, if you desire only a partial rollback, position the audit trail (using positioning commands such as LOCATE, FIND, FORWARD, BACK, NEXT, UP) at the point where the rollback is to terminate. The rollback will start at the end of the audit trail and proceed backwards up to and including the entry at the current audit trail position.
4. Enter the AUDITRL command.
5. Switch back to viewing the text as follows:

```
=> screen aud,off
```

6. Verify that the status of the text is what you intended. If it is, SAVE the member. Otherwise, enter "END NOSAVE".

Note that the rollback process itself is audited. So, if necessary, the rollback itself can be rolled back.

Use in a Procedure

AUDITRL normally returns OK. (See SIBRETC D.)

AUDITSM

Use AUDITSM to create a session displaying a summary of the changes to a member based upon its audit trail.

AUDITSM may also be entered as AUDSM.

Optional Operands	
MEM	is the member name for which changes are to be summarized. If MEM is not specified, the "last referenced member" is used.

A member's audit trail provides a detailed description of updates against a member. AUDITSM, on the other hand, uses the audit trail as input and produces as output a summarization of those changes. AUDITSM shows only net changes. For example, if 5 lines were inserted, and if those lines were subsequently modified, AUDITSM shows that 5 lines were inserted but shows the modified text. Since the lines were new, the fact that they got the way they are through modification is irrelevant.

AUDITSM can be used to perform "version" auditing if a new version always starts with a cleared audit trail. Procedures involving CHECKIN and CHECKOUT could be used to enforce this.

AUDITSM sets the "last referenced member".

When Valid

The user must have LIST access level for the MEM library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Member not found.
SC	Inadequate access level.

AUDITSM sets the TXM variables to the attributes of MEM.

Example

Summarize changes for member POLICIES:

AUDIT display before AUDITSM command

```

=> auditsm policies
LIST  CHKG.POLICIES                      SESS=A 1( 1) LINE=    0(   72)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==== -- TOP OF AUDIT --
*==== .EDIT  DATE=06/16/1996, TIME=13:40:53, USER=MWD , TERM=LL01
*==== .MOD   LOC=      3
*==== Generally, this account is one which may be used by more than one
*==== GENERALLY, A JOINT ACCOUNT IS ONE WHICH MAY BE USED BY MORE THAN ONE
*==== .MOD   LOC=      3
*==== .MOD   LOC=      7
*==== other owner. If one joint owner dies, the account becomes the
*==== OTHER DEPOSITOR. IF ONE JOINT DEPOSITOR DIES, THE ACCOUNT BECOMES THE
*==== .MOD   LOC=      7
*==== .MOD   LOC=      8
*==== property of the remaining joint owner(s). This account is commonly
*==== PROPERTY OF THE REMAINING JOINT DEPOSITOR(S). THIS TYPE OF ACCOUNT IS
*==== .MOD   LOC=      8
*==== .MOD   LOC=      9
*==== referred to as a "joint account with right of survivorship." It is
*==== LEGALLY REFERRED TO AS A "JOINT ACCOUNT WITH RIGHT OF SURVIVORSHIP." IT
*==== .MOD   LOC=      9

```

AUDITSM display

```

=>
DISP  -> auditsm                      SESS=A 2( 2) LINE=    0(   21)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==== -- TOP OF DISPLAY --
*==== -----
*==== AUDIT SUMMARY, MEMBER = ED51A.POLICIES
*==== -----
*==== -> MODIFY  ORIG REF= LINE      3, CUR REF= LINE      3, LINES=    2
*====
*==== Generally, this account is one which may be used by more than one
*==== person as if it were each depositor's own account. Each person who
*==== ->
*==== Generally, a joint account is one which may be used by more than one
*==== depositor as if it were the depositor's own account. Each person who
*====
*==== -> MODIFY  ORIG REF= LINE      7, CUR REF= LINE      7, LINES=    4
*====
*==== other owner. If one joint owner dies, the account becomes the
*==== property of the remaining joint owner(s). This account is commonly
*==== referred to as a "joint account with right of survivorship." It is
*==== your responsibility to notify the bank of the death of an account
*==== ->
*==== other depositor. If one joint depositor dies, the account becomes the

```

BACK

Use BACK to position the current session up (toward the top) a specified number of logical screens.

BACK may also be entered as BA or BACKWARD.

Optional Operands	
FCT	specifies the number of logical screens to move the session position by. If FCT is not specified, your session will be positioned backward one screen.

A logical screen is exactly the number of lines of text shown in the current text display area -- it varies according to terminal size and whether SCREEN SPLIT is ON.

Use in a Procedure

BACK normally returns OK. (See SIBRETCD.)

Examples

Position back 1 screen:

```
=> back
```

Position back 4 screens:

```
=> ba 4
```

BLANK

Use BLANK to replace current session text with blanks in a specified column range for a specified number of lines starting with the current line.

BLANK may also be entered as BL.

Optional Operands	
ZONE	is the column range to blank. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to blank columns 11 to 20, specify ZONE as "11-20". ZONE=5 is the same as ZONE=5-5, ZONE=5-* is the same as ZONE=5-253 and ZONE=-5 is the same as ZONE=1-5. If ZONE is not specified, the current session zone is blanked (see the ZONE operand of the DEFINE, ALTER, and SESS commands.)
FCT	specifies the number of lines to blank. If FCT is specified as an asterisk (*), all of the lines through the end of the session will be blanked. If FCT is not specified, only the current line will be blanked.

KEEP blanks columns except in a specified column range.

Use in a Procedure

BLANK normally returns OK. (See SIBRETC D.)

Example

EDIT session before BLANK command

```
=> blank 21-253,*
EDIT CFF.TEMP                                SESS=A 1( 1) LINE=      0( 72)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
***** -- TOP OF MEMBER --
***** LST BIMWNDXX 1427 3 H A      5      1 1      BM01
***** LST BIMWND  1446 3 H A    551    17 1
***** LST BIMWNC  1620 3 H A  4,472   109 1
***** LST BIMWNX   1448 3 H A    20     1 1      BM01
```

EDIT session after BLANK command

```
=>
EDIT CFF.TEMP                                SESS=A 1( 1) LINE=      0( 72)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
***** -- TOP OF MEMBER --
***** LST BIMWNDXX 1427
***** LST BIMWND  1446
***** LST BIMWNC  1620
***** LST BIMWNX   1448
```

BOTTOM

Use BOTTOM to position the current session at the furthest forward (down-most) logical screen.

BOTTOM may also be entered as BOT, BO, or B.

BOTTOM has no operands.

Note that this command does not position at the last line, but at the last screen. The last line of the session becomes the last line on the screen.

Use in a Procedure

BOTTOM normally returns OK. (See SIBRETC D.)

To go to the bottom of a session prior to inserting new lines or performing LOCATEU, FINDUP, or NFINDUP, use NEXT 999999. (Note that LOCATEU, FINDUP, or NFINDUP cannot find an instance of the specified string ON the last line because it is impossible to position AFTER the last line.)

Examples

Position at the last screen:

```
=> bottom
```

Position at the last line (as opposed to the last screen):

```
=> bot;forward
```

CANCEL (MVS only)

Use CANCEL to terminate a JES job.

CANCEL may also be entered as CANC.

Optional Operands	
JOB	selects the JES job to be terminated. If JOB contains letters, it specifies a jobname label from the Job Control Language JOB statement. If JOB is entirely digits, it specifies a job number assigned by JES. If not specified, the job from "last referenced JES data sets" will be used.
OPT	If specified as D or DUMP, a memory dump of the current step will be produced. If not specified, a dump will not take place.

To usefully cancel a JES job, the job must not have completed processing.

CANCEL does not delete the JES data sets associated with the job. Use PURGEP to delete a JES job and its data sets.

CANCEL sets "last referenced JES data sets".

When Valid

Check with your System Administrator to determine the restrictions on access to JES data sets at your site. The exit routine distributed with BIM-EDIT provides several restriction schemes your System Administrator can select among. (Use of and changes to the BIXPWQA exit routine are described in the System Reference Manual.)

Use in a Procedure

Return Codes:

OK	Successful.
NF	JES job not found.
*	Codes as defined by BIXPWQA exit routine.

CANCEL sets the JQE, JCT, and PDB variables to the attributes of the JES job cancelled.

Examples

Cancel the job just submitted:

```
=> cancel
```

Cancel job 7325, and produce a dump:

```
=> canc 7325,dump
```

CATAL (VSE version)

Use CATAL to copy the text of a BIM-EDIT member to a VSE sublibrary.

CATAL may also be entered as CAT.

Required Operands	
MEM	the name of the existing BIM-EDIT member to copy.
MEMD	the name and type of the VSE sublibrary member which is the destination for CATAL. The currently attached sublibrary (see ATTACHD) will be used unless the library and sublibrary are prefixed to the member name (i.e. library.sublib.member.type).

Optional Operands	
DATA	specifies whether a PROC type VSE member contains SYSIPT data. Can be "YES" or "NO". If unspecified, "NO" is assumed.
REPL	specifies whether a VSE member that already exists is to be replaced. Can be "YES" or "NO". If unspecified, "NO" is assumed.
EOD	specifies the two byte value to be used as end of data for the LIBR CATALOG request. EOD can be any two characters except comma or blank. If not specified, EOD defaults to X'FEDC'.

CATAL sets the "last referenced member" and the "last referenced VSE sublibrary member".

When Valid

BIM-EDIT must be running in its own partition and under VSE/SP release 2.1 or later. The user must have LIST access level for the MEM library, and DEF access level for the MEMD sublibrary.

Use in a Procedure

Return Codes:

- OK Successful.
- NF One of the following:
 - MEM member not found.
 - MEMD sublibrary not found.
- SC Inadequate access level.

CATAL sets the TXM variables to the attributes of MEM.

Examples

Catalog member PYRL007 from the current BIM-EDIT library as a PROC type member containing SYSIN data in the current VSE sublibrary:

```
=> catal pyrl007,pyrl007.proc,data=yes
```

Catalog member PAYMAST-LAYOUT from the BIM-EDIT PAYROLL library as PAYMAST, a C (COBOL) type member in the PAYROLL sublibrary of the PRODLIB VSE library:

```
=> cat payroll.paymast-layout,prodlib.payroll.paymast.c
```

CATAL (MVS version)

Use CATAL to copy the text of a BIM-EDIT member to a Partitioned Data Set member. CATAL may also be entered as CAT.

Required Operands	
MEM	the name of an existing BIM-EDIT member which is the source for CATAL. Specify lib.mem if you want to use a member from a BIM-EDIT library other than the currently attached one.
MEMD	the name of the PDS member which is the destination for CATAL. The currently attached PDS (see ATTACHD) will be used unless the PDS name is prefixed to the member name, that is, pdsname.member or pdsname(member).

CATAL sets the "last referenced member" and the "last referenced PDS member".

When Valid

The user must have LIST access level for the MEM library, and DEF access level for the MEMD sublibrary. MEMD must not already exist.

Use in a Procedure

Return Codes:

- OK Successful.
- NF One of the following:
 - MEM member not found.
 - MEMD PDS directory not found.
- SC Inadequate access level.

CATAL sets the TXM variables to the attributes of MEM.

Examples

Catalog member PYRL007 from the current BIM-EDIT library to the current PDS.

```
=> catal pyrl007,pyrl007
```

Catalog member PAYMAST-LAYOUT in the BIM-EDIT PAYROLL library as PAYMAST in the PRODLIB.PAYROLL PDS:

```
=> cat payroll.paymast-layout,prodlib.payroll.paymast
```

Catalog member 428 in the current BIM-EDIT library as PROG428 in the SYSLIB.PROBLEM PDS:

```
=> catal 428 syslib.problem(prog428)
```

CENTER

Use CENTER to center current session text within a specified column range for a specified number of lines starting with the current line.

CENTER may also be entered as CENT or CEN.

Optional Operands	
FCT	specifies the number of lines for which text is to be centered. If FCT is specified as an asterisk (*), text will be centered for all of the lines through the end of the session. If FCT is not specified, text will be centered for the current line only.
ZONE	is the column range in which to center text. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to center in columns 11 to 20, specify ZONE as "11-20". ZONE=5 is the same as ZONE=5-5, ZONE=5-* is the same as ZONE=5-253 and ZONE=-5 is the same as ZONE=1-5. If ZONE is not specified, text will be centered in the current session zone (see the ZONE operand of the DEFINE, ALTER, and SESS commands.)

CENTER is the command line equivalent of the LCA + command.

Use in a Procedure

CENTER normally returns OK. (See SIBRETCDD.)

Examples

Center text within column range 11 to 60 for 5 lines:

```
=> center 5,11-60
```

Center text within the current session zone for the current line:

```
=> cen
```

CHANGE

Use CHANGE to replace occurrences of a specified string of current session text characters with another specified string. Occurrences will be replaced in a specified column range for a specified number of lines starting with the current line.

CHANGE may also be entered as CH or C.

Required Operands	
OSTR	is the "old string" to be replaced. Up to 72 characters may be specified. If the value of OSTR begins with the backslash character (\), matching will use Extended Search Pattern rules, as described below.
NSTR	is the "new" replacement string. Up to 72 characters may be specified. Wherever OSTR is matched, NSTR will replace the characters which are matched.

Optional Operands	
FCT	specifies the number of lines to apply the change to. FCT may also be specified with an asterisk(*), which indicates all lines from the current line to end of session. If FCT is not specified, only the current line will be changed.
ZONE	is the column range in which the search and replacement will occur. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to limit the search to columns 11 to 20, specify ZONE as "11-20". ZONE=5 is the same as ZONE=5-5, ZONE=5-* is the same as ZONE=5-253 and ZONE=-5 is the same as ZONE=1-5. If ZONE is not specified, the current session zone is searched (see the ZONE operand of the DEFINE, ALTER, and SESS commands.)
SLMT	is the maximum column which will be used in shifting characters when OSTR and NSTR are not the same length. SLMT must not be less than the yyy column value used for ZONE. Default is the ZONE yyy value. When NSTR is longer than OSTR, characters are shifted right, and some will get deleted at the SLMT column. When NSTR is shorter than OSTR, characters are shifted left from the columns higher than the leftmost changed field on the line, and blanks are filled in at the SLMT column.
OCCUR	is a value which can be used to control the number of occurrences of an argument which will be changed on a line. The number of occurrences may be specified as a value from 1 to 9. A value of zero means that all occurrences of the argument on a line will be changed. The default is zero.

Upon completion of the CHANGE, the information line will display the number of lines scanned and the number of lines changed.

If session CASE is U, NSTR and OSTR will be translated to upper case letters before searching or replacing. If session CASE is M, NSTR and OSTR are used "as is". Session text is never translated before comparison as it is with the FIND, FINDUP, LOCATE, LOCATEU, NFIND, NFINDUP, QUALIFY, and SCAN commands. For example, if session CASE is U and you enter

```
=> change 'hello' 'good bye'
```

only instances of "HELLO" in upper case will be changed to "GOOD BYE" in upper case. For the same command, if session CASE is M, only instances of "hello" in lower case will be changed to "good bye" in lower case. Under some circumstances, you may want to use SESS CASE= before using CHANGE.

If the value of OSTR begins with a backslash (\), BIM-EDIT treats OSTR as an Extended Search Pattern, with certain characters having special meanings:

- ? matches any single character
- * matches any sequence of zero or more characters
- @ matches zero or more instances of the character that follows the @
- ! matches the beginning of the zone
- < matches the beginning of the zone or any non-alphanumeric character
- > matches the end of the zone or any non-alphanumeric character
- | separates two patterns, EITHER of which may match for a successful search
- + separates two patterns, BOTH of which must match for a successful search
- ~ (NOT) reverses line-by-line success/failure of the pattern following it
- % marks the position on the line where the match is considered to occur
- \ suppresses the special meaning of the character that follows the \

% in an Extended Search Pattern has special relevance to CHANGE. If there are two % characters in OSTR, they indicate where replacement by NSTR will start and end. If there is only one %, replacement runs from the % to the last non-pattern character in OSTR. Of course, if there aren't any % characters, replacement will be from the first to the last non-pattern characters in OSTR.

See Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual for more information about Extended Search Patterns.

CHANGE should be used with care when it will affect many lines. It is quite easy to write a CHANGE which will have undesired effects, for example

```
=> change "the" "a" *
```

will not only change the word "the" to the word "a", it will also change the word "other" to the word "oar" and the word "there" to the word "are".

You can protect against a runaway CHANGE in several ways:

1. SAVE and restart the EDIT session before entering the CHANGE. That way, if undesirable changes take place, you can END NOSAVE to undo the CHANGE.
2. Use Extended Search Pattern characters to further qualify the CHANGE. For example, the command above could be written more safely as

```
=> change "\<the>" "a" *
```

3. Use QUALIFY to preview the lines which will be changed.
4. Use a combination of LOCATE and a one-line CHANGE to view each line after it is changed. For the above example:

```
=> &locate "the" case=m;change "the" "a"
```

Each time you hit ENTER, a new line will be found, changed and displayed until the LOCATE fails because it can't find any additional matches.

Use in a Procedure

Return Codes:

OK Successful.
SV OSTR length zero or SLMT less than ZONE limit.

Example

EDIT session before CHANGE command

```
=> change key,xkey,*
EDIT  4216.RM0560                      SESS=A 1( 1) LINE=      8(   76)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*      IF KEY < LIMIT THEN
*====*      DO;
*====*          KEY = KEY + 1;
*====*          SAVE KEY = KEY * 4;
*====*      END;
*====*      CFNAME = '';
*====*      CFADDR1 = '';
*====*      CFADDR2 = '';
*====*      CFCITY = '';
*====*      CFSTATE = '';
*====*      CFPHONE = '';
*====*      CFBAL = 0;
*====*      IF OPT = '1' THEN
*====*      DO;
*====*          CFLIST = 0;
*====*          CFSTM = 0;
*====*          KEY = KEY + 1;
*====*      END;
```

EDIT session after CHANGE command

```
=>
## SCANNED 69 LINES, UPDATED 4 LINES ##
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==*      IF XKEY < LIMIT THEN
*==*      DO;
*==*          XKEY = XKEY + 1;
*==*          SAVE XKEY = XKEY * 4;
*==*      END;
*==*      CFNAME = '';
*==*      CFADDR1 = '';
*==*      CFADDR2 = '';
*==*      CFCITY = '';
*==*      CFSTATE = '';
*==*      CFPHONE = '';
*==*      CFBAL = 0;
*==*      IF OPT = '1' THEN
*==*      DO;
*==*          CFLIST = 0;
*==*          CFSTM = 0;
*==*          XKEY = XKEY + 1;
*==*      END;
```

CHECKASN

Use CHECKASN to move the working (slave) copy of a master member to another library and/or to assign the working (slave) copy to a different user.

CHECKASN may also be entered as CHKA.

Required Operands	
MEM	is either the master member or the slave member in an existing checkout relationship.
LIB	is the library to which the slave member is to be moved.
USER	is the user to which the slave member is to be assigned.

The slave member will be moved to the LIB library or assigned to the USER user.

The master member is updated to indicate that it is checked out to the LIB library, and the USER user. These can be displayed with the INQUIRE command.

CHECKASN sets "last referenced member" to the slave member.

When Valid

The user must have DEF access level to the master member library and DEF access level to the LIB library.

Use in a Procedure

Return Codes:

OK	Successful.
CK	Member is not part of a checkout relationship.
DP	Member already exists in the LIB library.
NF	One of the following: <ul style="list-style-type: none">• User not found.• MEM member not found.• LIB library not found.
SC	Inadequate access level.

CHECKASN sets the TXM variables to the attributes of the slave member.

Examples

Move the slave member OM20.OMRPWR3 to library MWD and assign the slave member to user MWD.

```
=> checkasn om20.omrpwr3,mwd,mwd
```

Move the slave member associated with the master member BIREXIO to library ED35 and assign the slave member to user LSL.

```
=> chka birexio,ed35,ls1
```


CHECKIN

Use CHECKIN to replace the text of a master member with the text of its working (slave) copy.

CHECKIN may also be entered as CHKI.

Required Operands	
MEM	specifies either the master or the slave member in a checkout relationship.

BIM-EDIT will replace the master with the slave, then purge the slave. If you want to delete the slave without updating the master, use the CHECKPUR command.

After being checked in, the member is again in a position to be checked out. When the ARCHIVE feature is installed the CHECKIN process will create a new -1 generation level in the \$SIT.GEN library. All other generations are moved up one level (-1 becomes -2, etc.), with the oldest being moved to the \$SIT.ARCHIVE library.

CHECKIN sets "last referenced member" to the master member.

When Valid

If the member is checked out to the user requesting the checkin, the user must have EDIT access level to the master member library. Otherwise, the user must have DEF access level to the master member library. In any case, if MMPAUCTL is set to 1, 'ADM' command security is required.

Use in a Procedure

Return Codes:

OK	Successful.
CK	Member is not part of a checkout relationship.
ED	Member is being edited.
LI	List session(s) active.
NF	Member not found.
SC	Inadequate access level.

CHECKIN sets the TXM variables to the attributes of the master member.

Example

Checkin member GJB.OMRPWR3:

```
=> checkin gjb.omrpwr3
```

CHECKOUT

Use CHECKOUT to create the working (slave) copy of a master member.

CHECKOUT may also be entered as CHKO.

Required Operands	
MEM	is the member that will be the master in the checkout relationship. This member must have CHECK=ON (see the DEFINE and ALTER commands) and must not already be checked out.

Optional Operands	
LIB	is the library in which to create the slave member. LIB must not be the same as the MEM library. If LIB is not entered, the currently attached library is used.
USER	is the user to check out the member to. If USER is not specified, the logon user ID will be used.
GEN	when the ARCHIVE feature is installed this is the generation level of the member to be checked out. If GEN is not specified, the current master is checked out.

A slave copy of the MEM member will be created and stored in the LIB library under the MEM member name.

This command is intended for use in a situation where a library of master (production) members is to be maintained in a controlled manner. The checkout/checkin mechanism assures that all modifications are made to slave copies of the master and that only one slave of a master exists at a time. If many people have LIST access level to the master library, they can all CHECKOUT slave copies. If only one person has DEF access level, that person controls all CHECKINs.

If an independent copy of a master is desired, use the COPY command.

The slave member is created with all of the master member attributes, with the exception that is flagged as the slave in the checkout relationship. Both the member text and the audit trail (if it exists) are copied. A slave in a checkout relationship can be edited only by the user to whom the member is checked out, but that user can edit the slave even if it is in a library to which the user does not have access. A slave cannot be purged by the PURGE command, nor altered by the ALTER command. A slave can be eliminated either by the CHECKIN command or the CHECKPUR command.

The master member is updated to indicate that it is checked out to the slave library, and the checkout user ID. These can be displayed with the INQUIRE command.

INCLUDE commands behave slightly differently when they are contained in the slave member of a checkout relationship. For purposes of INCLUDE expansion, the member is assumed to be part of the library where the master resides. Therefore, if the full member name (lib.mem) is not specified with the INCLUDE command, BIM-EDIT will

include the member from the master library, not the slave library. To INCLUDE a member from the slave library, specify the full member name.

When the ARCHIVE feature is installed and GEN is specified, the current source is checked out and then edited to reverse all changes to obtain the requested generation. GEN is specified as "-n" where n is a number from 1 to 9 depending on the number of generations maintained by the system (see RECOVER). When GEN is specified the user field of both slave and master are altered to indicate the level checked out.

CHECKOUT sets "last referenced member" to the slave member.

When Valid

If USER is entered and not equal to the logon user ID, the user must have DEF access level to MEM library. Otherwise, the user must have LIST access level to the MEM library. In any event, the user must have DEF access level to the LIB library. MEM must have CHECK=ON (see the DEFINE and ALTER commands) and must not already be checked out.

Use in a Procedure

Return Codes:

OK	Successful.
CK	One of the following: <ul style="list-style-type: none">• Member is already checked out.• Member is not under checkout control.
DP	Member already exists in the LIB library.
NF	One of the following: <ul style="list-style-type: none">• Υσερ νοτ φουνδ.• ΜΕΜ μεμβερ νοτ φουνδ.• ΛΙΒ λιβραριυ νοτ φουνδ.
RG	One of the following (ARCHIVE feature only) <ul style="list-style-type: none">• νο πρεπιουσ γενερατιονσ φορ μεμβερ.• ρεγεγερατιον σουρχε ηασ βεεν χορρυπτεδ.
SC	Inadequate access level.

CHECKOUT sets the TXM variables to the attributes of the slave member.

Examples

Checkout member OM20.OMRPWR3 to library GJB for user GJB:

```
=> checkout om20.omrpwr3,gjb,gjb
```

Checkout member OM20.OMREXIO for ourself. Place it in the currently attached library:

```
=> chko om20.omrexio
```

CHECKPUR

Use CHECKPUR to delete the working (slave) copy of a master member without updating the master member.

Required Operands	
MEM	specifies either the master or the slave member in a checkout relationship.
FRC	specifies whether the checkout relationship is to be severed even though it appears to be incomplete. The FRC option must be entered as FORCE and requires ADM security level access.

BIM-EDIT will sever the checkout relationship, then purge the slave member. The master is then again eligible for checkout.

CHECKPUR sets "last referenced member" to the master member.

If your System Administrator has set predefined variable MMPPGCTL to "1", members can be retrieved for a short period of time after an erroneous PURGE. Otherwise, the only possible way to retrieve a member after a PURGE is to use RESTORE from a backup tape.

When Valid

If the member is not checked out to the user requesting CHECKPUR, the user must have DEF access level for the master member library.

Use in a Procedure

Return Codes:

OK	Successful.
CK	Member is not part of a checkout relationship.
ED	Member is being edited.
NF	Member not found.
SC	Inadequate access level.

CHECKPUR sets the TXM variables to the attributes of the master member.

Example

Check purge member GJB.OMRPWR3:

```
=> checkpur gjb.omrpwr3
```

COMPARE

Use COMPARE to compare the session lines of the two sessions currently being displayed in Split Screen mode.

COMPARE may also be entered as CMPR.

Optional Operands	
FCT	specifies the number of lines in each session to be compared. If FCT is specified as an asterisk (*), or is not specified, all of the lines, starting with the current line in each session, through the end of each session will be compared.
ZONET	is the column range to be compared for the session lines in the top logical screen. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to compare only the columns 11 to 20, specify ZONET as "11-20". ZONET=5 is the same as ZONET=5-5, ZONET=5-* is the same as ZONET=5-253 and ZONET=-5 is the same as ZONET=1-5. If ZONET is not specified, the current session zone is used. (see the ZONE operand of the DEFINE, ALTER, and SESS commands.)
ZONEB	is the column range to be compared for the session lines in the bottom logical screen. The format is the same as the ZONET operand. If ZONEB is not specified, the current session zone is used.
CASE	specifies whether upper/lower case should be considered when comparing the session lines. If specified as "U", case is ignored ("the" matches "The"). If specified as "M", both case and letters must match. If CASE is not specified, "M" is assumed.

COMPARE begins comparing session lines at the current line of each of the sessions currently being displayed in the split screen.

Any type of session can be compared to any other type of session. Two portions of the same session can also be compared by limited the compare with the FCT operand.

If the session lines of both sessions are the same, the display is left at the same position and a message is displayed on line 2 of the screen indicating the success of the COMPARE.

If the session lines of both sessions are not the same, each session is positioned to the first line containing a difference, and a message is displayed on line 2 of the screen indicating the column of the first difference.

If one of the sessions has fewer lines than the other, and all lines match up to the end of the shorter session, each session is positioned to the last matching line, and a message is displayed on line 2 indicating this condition.

If the widths of the ZONES of the two sessions are not the same, the comparison is performed as if the contents of the narrower ZONE is extended on the right with blanks to equal the width of the wider ZONE.

After the comparison ends, due to a mismatch, you can manually reposition each session so that the current line of each session is a new comparison begin point, and reenter the COMPARE command to continue comparing lines. (Each entry of the COMPARE command is treated as a new command, any operands specified must be respecified each time the command is entered.)

When Valid

COMPARE can only be used as an online command.

Examples

The following sequence of commands will compare members LIB1.MEMBER1 and LIB1.MEMBER2:

```
=> list lib1.member1
=> screen split on
=> list lib1.member2
=> compare
```

COMPILE

Use COMPILE to submit a member for batch processing to be compiled as a program. COMPILE may also be entered as COMP.

Optional Operands	
MEM	is the member to be compiled. If MEM is not specified, the "last referenced member" is used.
PSWD	If the member is password protected, the password must be entered before the compile request is allowed. The password may be up to 8 characters in length consisting of any characters.

COMPILE is often invoked through the PROCESS command, which in turn is invoked from either the command line or the LCA S command. See the PROCESS command documentation for more information.

The type of compile is determined by the TYPE and ATTR fields for the member. The operating system libraries to search and/or catalog to are determined by the LIBDEF field for the member. TYPE, ATTR, and LIBDEF should be set for the member prior to using COMPILE by using DEFINE, FDEFINE or ALTER. For information on what to set these values to, contact your System Administrator.

COMPILE sets the "last referenced member".

When Valid

The user must have LIST access level for the MEM library.

Use in a Procedure

Return Codes:

OK	Successful.
ED	Member is being edited.
NF	Member not found.
SC	Inadequate access level.

COMPILE sets the TXM variables to the attributes of MEM.

COMPILE is implemented as the system procedure BIPCOMP.

Examples

Compile the program held in member OMREXIO:

```
=> compile omrexio
```

Save and compile the current EDIT session:

```
=> save;comp
```

(You may wish to set a PF key to this sequence.)

CONSOLEI

Use CONSOLEI to write a specified line/command to the operating system console.

CONSOLEI may also be entered as CNSLI or CONSI.

Required Operands	
LINE	is the line to display on the operating system console.

When Valid

The CONSOLEI command is used to either send an operating system command to the system console, or to simply write a message to the system console, depending on the security level of the user.

The system variable MMPCNSLI may be set by the system administrator to the value of the command security field for type of user (see SIBSECCM). If a user's command security field value is equal to or greater than this value he may send commands to the operating system console.

Use in a Procedure

CONSOLEI normally returns OK. (See SIBRETCD.)

Example

Ask the operator to print a job:

```
=> consolei 'jack, please print xmr0235.  thanks, jim'
```


COPY

Use COPY to create a copy of an existing member.

Required Operands	
SRC	is an already existing member.
DEST	is the name of the member to be created. A member of this name must not already exist.

Optional Operands	
PSWD	If the member is password protected, the password must be entered. The password may be up to 8 characters in length consisting of any characters. A password protected member that is copied will have the same password as the original member.
REPL	is used to specify whether the DEST member is to be replaced if a member by this name already exists. The default is "NO". If REPL=YES is specified and the DEST member does not exist the COPY will be completed but the completion message will display the fact that the DEST member was not found. It is not possible to replace the following: <ul style="list-style-type: none"> • a master member (i.e. check status is on) • a member that is part of a checkout relationship • a member that is password protected
ORIGDATE	is used to specify whether the original create date/time and current update date/time are to be copied from the original source (SRC) member to the destination (DEST) member. Can be specified as "YES" or "NO". The default is "NO". If "NO" is specified, or allowed to default, the newly created copy will use the current date/time as its create and update values.

The FCOPY command may provide a more convenient way to copy a member.

A copy of the source member will be created under the destination name. COPY and DEFINE are the only commands that create new members. Both the text and the member attributes are copied. The source member is unaffected by the COPY.

If an asterisk (*) is entered as part of DEST, the asterisk is replaced with the SRC member name. This facilitates copying a member from one library to another or making a copy of a member to a suffixed or prefixed name without keying the member name twice (see Examples).

To move a member from one library to another, see the MOVE, RENAME or FRENAME commands.

COPY sets "last referenced member" to the DEST member.

When Valid

The user must have LIST access level for the SRC library and DEF access level for the DEST library.

Use in a Procedure

Return Codes:

OK	Successful.
DP	DEST member already exists.
NF	Either the SRC member was not found or the DEST library was not found.
SC	Inadequate access level.

COPY sets the TXM variables to the attributes of DEST.

Examples

Create a backup copy of member SYREXIO:

```
=> copy syrexio,syrexio-back -or- => copy syrexio,*-back
```

Create SYCREN using SYCLIST as a skeleton:

```
=> copy syclist,sycren
```

Copy OMRPWR3 to library OM11C:

```
=> copy omrpwr3,om11c.omrpwr3 -or- => copy omrpwr3,om11c.*
```

COPYD (VSE version)

Use COPYD to create a copy of an existing VSE member.

Required Operands	
SRC	is an already existing member. The currently attached sublibrary (see ATTACHD) will be used unless the library and sublibrary are prefixed to the member name (i.e. lib.sublib.member.type).
DEST	is the name of the member to be created. If the destination member name already exists, the REPL operand may be used to cause it to be overwritten. The currently attached sublibrary (see ATTACHD) will be used unless the library and sublibrary are prefixed to the member name (i.e. lib.sublib.member.type).

Optional Operands	
REPL	If the member name already exists in the destination library, a value of "YES" may be entered to cause the member to be overwritten. If omitted, the default is "NO".

A copy of the source member will be created under the destination name. COPYD and DEFINED are the only commands that create new VSE members. Both the text and the member attributes are copied. The source member is unaffected by the COPYD.

To move a member from one library to another, see the RENAMED command.

COPYD sets the "last referenced VSE sublibrary member".

When Valid

The user must have LIST access level for the SRC library and DEF access level for the DEST library.

Use in a Procedure

Return Codes:

OK	Successful.
DP	DEST member already exists.
NF	Either the SRC member was not found or the DEST library was not found.
SC	Inadequate access level.

Examples

Create a backup copy of member \$ASIPROC:

```
=> copyd $asiproc.proc $asiproc.backup
```

COPYD (MVS version)

Use COPYD to create a copy of an existing PDS member.

Required Operands	
SRC	is an already existing PDS member. The currently attached PDS (see ATTACHD) will be used unless the PDS name is prefixed to the member name, that is, pds.member or pds(member).
DEST	is the name of the member to be created. A member of this name must not already exist. The currently attached PDS (see ATTACHD) will be used unless the PDS name is prefixed to the member name, that is, pds.member or pds(member).

A copy of the source member will be created under the destination name. COPYD and DEFINED are the only commands that create new PDS members. Both the text and the member attributes are copied. The source member is unaffected by the COPYD.

COPYD sets the "last referenced PDS member".

When Valid

The user must have LIST access level for the SRC library and DEF access level for the DEST library.

Use in a Procedure

Return Codes:

OK	Successful.
DP	DEST member already exists.
NF	Either the SRC member was not found or the DEST library was not found.
SC	Inadequate access level.

Examples

Copy member OLDMEM to NEWMEM in the current PDS:

```
=> copyd oldmem newmem
```

Copy member PYRLMNTH from the current PDS to the PRODLIB.PAYROLL PDS:

```
=> copyd pyrlmnth prodlib.payroll.pyrlmnth
```

CURSOR

Use CURSOR to position the cursor on the current line in the column where the most recent EXAMINE, FIND, FINDUP, LOCATE, LOCATEU, NFIND or NFINDUP command found a match.

CURSOR can also be entered as CURS or CUR.

CURSOR has no operands.

Use in a Procedure

CURSOR normally returns OK. (See SIBRETCO.)

CURSOR positions to the column held in the SSDCOL1 (or SSDCOL2) predefined variable. The EXAMINE, FIND, FINDUP, LOCATE, LOCATEU, NFIND and NFINDUP commands set SSDCOL1 (or SSDCOL2).

Example

Locate each occurrence of string XLIB. For each occurrence, position the cursor at the string to allow easy update:

```
=> &locate xlib;cursor
```

DA (VSE version)

Use DA to display the operating system's processing status. When DA is invoked online, it creates a DISP session for the results.

Optional Operand	
ORDER	specifies the order of the partitions displayed. If specified as "PRTY", the partitions will be displayed in priority sequence, with the highest priority at the top and the lowest priority at the bottom. Static partitions that are not in use will not be displayed. If omitted, all Static Partitions will be displayed in PIK sequence, followed by all active Dynamic Partitions, in the order they appear in the DTR\$DYNx table.

Display sessions such as that produced by the DA command can be updated or "refreshed" by the REFRESH command. (You may find it convenient to assign the REFRESH command to a PF key.)

Use in a Procedure

DA normally returns OK. (See SIBRETC D.)

If DA is invoked in batch utility, the display destination is controlled by predefined variable SIBOUTPT.

Example

The following is an example of the DA output.

```
=>
DISP -> da                                SESS=A 1( 2) LINE=      0(      18)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----|-----
-- TOP OF DISPLAY --
                                DATE 06/28/2001  TIME 17:18:42
RUN  CODE  JOB NAME  START  DURATION  PHASE  START  DURATION  CPUTIME  STARTIO  STEP
BG
FB
FA WAIT  BIJEDITT  15:07:27  15:40:36  BIMEDIT  15:07:26  15:40:36    14.68    2,122
F9 READY BIM-EDIT  06:01:28  24:46:35  BIMEDIT  06:01:27          3172.18  412,043
F8 WAIT  BIMWNDOW  14:48:21    4 DAYS  BIMWN45E  14:48:21          2464.93    4,593
F7
F6
F5 R=014 EDITBACK  06:20:36  00:27:26  BIMUTIL  06:20:36  00:27:26   175.22   52,699
F4 WAIT  CICSPROD  01:10:34  29:37:29  DFHSIP   01:10:41    1489.43    6,796
F3 WAIT  VTAM      03:55:10    20 DAYS  ISTINCVT  03:55:09   11630.98   1,478K
F2 WAIT  CICSSTEST  01:10:22  29:37:40  DFHSIP   01:10:48    3530.34   15,789
F1 WAIT  IPWPOWER  03:54:46    20 DAYS  POWERNET  03:54:45   12769.08   754,077
Y1 WAIT  BIMTMAN   06:54:34    8 DAYS  BIMTMAN   06:54:34    3232.52    5,825

PRTY F7,C=F5=F6=BG=FB,FA,G,Y,F2,F4,F9,F8,F3,F1
-- END OF DISPLAY --
```

(The relationship of the text entries in the "RUN CODE" column to the Task Information Block Task Status Flag values from which they are determined is provided in member \$SYS.DOC.RUNCODES for purists.)

A (K) following a value in the STARTIO field means that the start I/O count has exceeded 999,999. The right most three digits have been dropped and the remaining digits shifted to the right.

The following is an example of the DA PRTY output.

```
=>
DISP -> da prty                                SESS= 1( 2) LINE=    0(   18)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----|-----
-- TOP OF DISPLAY --                          DATE 06/28/2001  TIME 17:18:42
      RUN  ----- JOB ----- STEP -----
      CODE JOB NAME  START  DURATION  PHASE    START  DURATION  CPUTIME  STARTIO
F1 WAIT  IPWPOWER  03:54:46  20 DAYS  POWERNET  03:54:45          12769.08  754,077
F3 WAIT  VTAM      03:55:10  20 DAYS  ISTINCVT  03:55:09          11630.98   1,478K
F8 WAIT  BIMWNDOW  14:48:21    4 DAYS  BIMWN45E  14:48:21          2464.93    4,593
F9 READY BIM-EDIT  06:01:28  24:46:35 BIMEDIT  06:01:27          3172.18  412,043
F4 WAIT  CICSPROD  01:10:34  29:37:29 DFHSIP   01:10:41          1489.43    6,796
F2 WAIT  CICSTEST  01:10:22  29:37:40 DFHSIP   01:10:48          3530.34   15,789
Y1 WAIT  BIMTMAN   06:54:34    8 DAYS  BIMTMAN   06:54:34          3232.52    5,825
FA WAIT  BIJEDITT  15:07:27  15:40:36 BIMEDIT  15:07:26  15:40:36    14.68    2,122
F5 R=014 EDITBACK  06:20:36  00:27:26 BIMUTIL  06:20:36  00:27:26   175.22   52,699

PRTY F7,C=F5=F6=BG=FB,FA,G,Y,F2,F4,F9,F8,F3,F1
-- END OF DISPLAY --
```

DA (MVS version)

Use DA to display the operating system's processing status. When DA is invoked online, it creates a DISP session for the results.

Optional Operands	
USER	is site-defined. USER can be up to 16 characters in length. The exit routine BIXPWQA is provided addressability to this value.

Display sessions such as that produced by the DA command can be updated or "refreshed" by the REFRESH command. (You may find it convenient to assign the REFRESH command to a PF key.)

When Valid

Check with your System Administrator to determine the restrictions on access to processing status at your site. The exit routine distributed with BIM-EDIT provides several restriction schemes your System Administrator can select among. (Use of and changes to the BIXPWQA exit routine are described in the System Reference Manual.)

Use in a Procedure

DA normally returns OK. (See SIBRETCDD.)

Example

The following is an example of the DA output.

```
=>
DISP  -> da                                SESS=A 1( 1) LINE=    0(    9)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----|-----
*====* -- TOP OF DISPLAY --
*====*
*====* ----- JOB ---- T C ASID   STEP      PROC   POS   REAL   EXCP   CPU
*====*      NAME  NUMBER Y L          STEP      (K)   COUNT  TIME
*====* -----
*====* BIM001B   8628 J I    32 BIMEDIT      N/S 1220K   69,258   7.25
*====* BIM001Z   8824 J I    29 ASM          IN   720K    8,725    4.34
*====*
*====* -- END OF DISPLAY --
```


DEFINE

Use DEFINE to create a new BIM-EDIT member.

DEFINE may also be entered as DEF.

Required Operands	
MEM	<p>is the member to be defined. MEM can be up to 16 characters in length, composed of the letters A-Z, the numerals 0-9, and the special characters dash (-), dollar sign (\$), pound sign (#), and the underscore (_). (The libname.memname format can also be used. In that case, MEM is up to 33 characters in length: 16 for libname, 1 for period, 16 for memname.)</p> <p>While BIM-EDIT supports the underscore (_) in the member name, it is not supported by all of the subsystems that BIM-EDIT interfaces with (for example, POWER). Care should be used when deciding which member names may contain an underscore(s).</p>

Optional Operands	
TYPE	<p>is the member type, which determines the initial attributes and contents of the new member. Member attributes from TYPE can be overridden by using other operands on the DEFINE. If TYPE is enclosed in parentheses (), the member will not have any initial contents. TYPE can be up to 8 characters in length. If TYPE is not specified, the type defined for model \$DFL is used.</p> <p>For program source members, TYPE is also typically used by the COMPILE command to determine which compiler to create Job Control Language for.</p> <p>BIM-EDIT, as distributed, provides the following TYPES:</p> <p>\$DFL Default template. ASM Assembler program source. COBOL COBOL program source. DATA Data. FORT FORTRAN program source. JCL DOS job streams. PLI PLI program source. PROC Procedure. RPG RPG program source. TEXT Textual material, such as documentation.</p> <p>Your System Administrator has probably added or deleted TYPES from the above list.</p> <p>TYPE is implemented by copying the attributes and text of a "template member" of name TYPE in the BIM-EDIT Site or</p>

Optional Operands	
	System model library to the new member.

Optional Operands (continued)	
TITLE	is a comment about or description of the member. TITLE can be up to 40 characters in length.
ATTR	is an attribute with site-defined usage -- check with your System Administrator. For program source members, ATTR is typically used by the COMPILE command to determine the nature of the compile. ATTR can be up to 8 characters in length.
AUDIT	Specify "ON" if member auditing is desired. "OFF" suppresses member auditing. Member auditing allows you to view and undo all modifications to member text, but it also uses considerable disk space and some additional processing time. If your System Administrator has set predefined variable MMPAUCTL to "1", only the System Administrator can DEFINE AUDIT=OFF. For more discussion, see the BIM-EDIT User Reference Manual, Chapter 6, Advanced Techniques.
CASE	Specify "U" if updated text lines are to be translated to upper case. Specify "M" if no translation is to occur. Only new and updated lines are affected by CASE.
CHECK	specifies whether member is under checkout\checkin control. Specify "ON" if editing can be done only on a checked out copy. Specify "OFF" if member can be edited directly. For more discussion, see BIM-EDIT User Reference Manual Chapter 6, Advanced Techniques.
LIBDEF	is an attribute with site-defined usage -- check with your System Administrator. LIBDEF can be 8 characters in length. This field is typically used by the COMPILE command to determine how to build job control for a compile job stream.
NULLS	specifies whether the trailing blanks on a line are to be replaced with nulls when displaying the member. "ON" specifies that trailing blanks will be replaced, while "OFF" specifies that they will not. When NULLS is on, the 3270 terminal "insert mode" can be used. However if text is entered to the right of null character(s) on a line, the text ends up shifted left since the nulls are not transmitted to BIM-EDIT.
SEQ	specifies a column range where the RESEQ command places sequence numbers. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to place sequence numbers in columns 73 through 80, specify SEQ as "73-80". The SEQ range cannot be wider than 8 columns.

Optional Operands (continued)	
TCOL	specifies a list of columns used for tabbing. Up to 12 tab settings may be specified. They are entered in the format "cc-cc-cc". Each setting consists of one to three digits which specify the column where the text will be placed. Tabbing range is from 1 to 253. Tab columns must be entered in ascending order. For a more extensive discussion of tabbing, see Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual.
USER	is purely documentary. It can be set to any value. When a member is created, if USER is not specified, USER is set to the logon USER id. Typically, this value is used to determine ownership when culling junk members from the library. It can be up to 8 characters.
ZONE	specifies a column range used by the edit and search commands (such as CHANGE, CENTER, EXAMINE, FORMAT, LOCATE, QUALIFY, etc.) if not specified explicitly on the command. For example, the CHANGE command will change occurrences of one string to another within the current ZONE only. Enter ZONE in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to set the zone to columns 1 through 72, specify ZONE as "1-72". ZONE=5 is the same as ZONE=5-5, ZONE=5-* is the same as ZONE=5-253 and ZONE=-5 is the same as ZONE=1-5.
FCOL	specifies a column used by the FIND, FINDUP, NFIND, and NFINDUP commands if not specified explicitly on the command. Enter FCOL as a number from 1 to 253. For example, the FIND command scans forward for a line with a specified string starting in the FCOL column.
STAMP	Specify "ON" if member stamping is desired. "OFF" turns member stamping off. If your System Administrator has set predefined variable MMPSPCTL to "1", only the System Administrator can DEFINE STAMP=OFF. For more discussion, see Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual.
WRAP	Specify "ON" if BIM-EDIT/XP Edit sessions of this member are to use word-wrap mode. "OFF" turns word-wrap mode off. (This operand is relevant only when editing a BIM-EDIT member from a workstation using the BIM-EDIT/XP product.)

Optional Operands (continued)	
CDATE	Specify an optional create date. CDATE is entered either in the format "mm/dd/yy", "mm/dd/yyyy", "dd/mm/yy" or "dd/mm/yyyy" depending upon the date format set by the System Administrator. If the date is valid, the member will be defined with this date as the creation date. This parameter is valid only if the user has ADM command level security. This parameter is provided as a migration option when converting from other source editors. If omitted, the current system date will be used.
CTIME	Specify an optional create time. The format is HH:MM:SS. If the time is valid, the member will be defined with this time as the creation time. This parameter is valid only if the user has ADM command level security. If CDATE is not specified or CTIME omitted, the time will default to the current system time.
PSWD	Specify an optional member password if the member is to be password protected. The password may be up to 8 characters in length consisting of any character. If this field is omitted the member will not be password protected.
REPL	is used to specify whether the member is to be replaced if a member by this name already exists. The default is "NO". If REPL=YES is specified and the member does not exist the DEFINE will be completed but the completion message will display the fact that the member was not found. It is not possible to replace the following: <ul style="list-style-type: none"> • a master member (i.e. check status is on) • a member that is part of a checkout relationship • a member that is password protected
PGCTL	is used to specify specific PURGE options for this member. The PGCTL has four values, "NORMAL", "ALWAYS", "NEVER", or "LOCKED". <p>"NORMAL" specifies that the member will use the active system option for purge control. (See MMPPGCTL).</p> <p>"ALWAYS" specifies that the member will always be written to the \$SIT.PURGE library regardless of the MMPPGCTL setting.</p> <p>"NEVER" specifies that the member will never be written to the \$SIT.PURGE library when purged.</p> <p>"LOCKED" specifies that the member cannot be purged.</p> <p>The default is "NORMAL".</p>

(The FDEFINE command may be easier to use than the DEFINE command if a number of attributes must be specified. FDEFINE displays the default attributes for the specified member type and allows you to change the attributes by overtyping them.)

DEFINE sets the "last referenced member".

The attributes of a member can be displayed with the INQUIRE or FALTER commands.

When Valid

The user must have DEF access level for the MEM library.

Use in a Procedure

Return Codes:

OK	Successful.
DP	Member already exists.
NF	One of the following: <ul style="list-style-type: none">• Library not found.• TYPE is not valid.
SC	Inadequate access level.

DEFINE sets the TXM variables to the attributes of MEM. See Chapter 4, Predefined Variables, in the BIM-EDIT System Reference Manual or HELP TXM.

Examples

Define member GLM2300 as a COBOL member:

```
=> define GLM2300,cobol,title='print general ledger'
```

Define a data member named UMDAT with nulls set to off:

```
=> def umdat,data,nulls=off
```

Define the COBOL member ESAR030:

```
=> define esar030,cobol,tcol=8-12-16-40-56
```

DEFINED (VSE version)

Use DEFINED to create a new VSE sublibrary member.

DEFINED may also be entered as DEFD.

Required Operands	
MEMD	the name and type of the VSE sublibrary member. The currently attached sublibrary (see ATTACHD) will be used unless the library and sublibrary are prefixed to the member name (i.e. library.sublib.member.type).

Optional Operands	
DATA	specifies whether a PROC type VSE member contains SYSIPT data. Can be "YES" or "NO". If unspecified, "NO" is assumed.
EOD	specifies the two byte value to be used as end of data for the LIBR CATALOG request. EOD can be any two characters except comma or blank. If not specified, EOD defaults to X'FEDC'.

DEFINED is used to create a member prior to using the EDITD command. The member created will have one blank line. (A VSE sublibrary member cannot be created or saved with zero lines.)

The member type is used by the EDITD and LISTD commands to set the initial characteristics of the sessions they create. Follow any naming standards set by your System Administrator.

DEFINED sets the "last referenced VSE sublibrary member".

When Valid

BIM-EDIT must be running in its own partition and under VSE/SP release 2.1 or later. The user must have DEF access level for the MEMD library. The MEMD member must not already exist.

Use in a Procedure

Return Codes:

OK	Successful.
DP	MEMD already exists.
NF	MEMD sublibrary not found.
SC	Inadequate access level.

Examples

Create member P723178X, an A (assembly) type member in the current sublibrary:

```
=> defined p723178x.a
```

Create member PYRLMNTH as a PROC type member containing SYSIN data in the PAYROLL sublibrary of the PRODLIB VSE library:

```
=> defd prodlib.payroll.pyrlmnth.proc data=yes
```


DEFINED (MVS version)

Use DEFINED to create a new PDS member.

DEFINED may also be entered as DEFD.

Required Operands	
MEMD	the name of the PDS member. The currently attached PDS (see ATTACHD) will be used unless the PDS name is prefixed to the member name, that is, pdsname.member or pdsname(member).

DEFINED is used to create a member prior to using the EDITD command. The member created will have no text lines.

The last qualifier of the PDS name is used by the EDITD and LISTD commands to set the initial characteristics of the sessions they create. Follow any naming standards set by your System Administrator.

DEFINED sets the "last referenced PDS member".

When Valid

The user must have DEF access level for the MEMD library. The MEMD member must not already exist.

Use in a Procedure

Return Codes:

OK	Successful.
DP	MEMD already exists.
NF	MEMD PDS not found.
SC	Inadequate access level.

Examples

Create member P723178X in the current PDS:

```
=> defined p723178x
```

Create member PYRLMNTH in the PRODLIB.PAYROLL PDS:

```
=> defd prodlib.payroll.pyrlmnth
```

Create member PROG428 in the SYSLIB.PROBLEM PDS:

```
=> defd syslib.problem(prog428)
```

DEFINEL

Use DEFINEL to create a new library.

DEFINEL may also be entered as DEFL.

Required Operands	
LIB	is the library to be created. LIB can be up to 16 characters in length, composed of the letters A-Z, the numerals 0-9, and the special characters period (.), dash (-), dollar sign (\$), pound sign (#), and the underscore (_).

Optional Operands	
TITLE	is a title of, comment about, or description of the library. TITLE can be up to 40 characters.
USER	is purely documentary. If USER is not specified, it will be set to the logon user ID.

(The FDEFINEL command may be easier to use than the DEFINEL command.)

When Valid

The user must have DEFL access level for the library prefix of the LIB library (the part of the library name preceding an imbedded period). If LIB has no prefix, the user must have system-wide DEFL access.

Use in a Procedure

Return Codes:

OK	Successful.
DP	Library already exists.
SC	Inadequate access level.

DEFINEL sets the TXL variables to the attributes of the LIB library. See Chapter 4, Predefined Variables, in the BIM-EDIT System Reference Manual.

Examples

Define library 4216:

```
=> definel 4216
```

Define library OM20 with a title:

```
=> defl om20,'work library'
```

DELETE

Use DELETE to delete a specified number of current session text lines starting with the current line.

DELETE may also be entered as DEL.

Optional Operands	
FCT	specifies the number of lines to delete. If FCT is greater than the number of lines remaining or is specified as an asterisk (*), deletion will proceed to end of session. If FCT is not specified, only the current line will be deleted.

DELETE is the command line equivalent of the LCA D command.

Use in a Procedure

DELETE normally returns OK. (See SIBRETCOD.)

Example

EDIT session before DELETE command

```
=> delete 3
EDIT  EMX3.EMSA200                      SESS=A 1( 1) LINE=      1(      6)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*      DCL
*====*      CF INV BAL                    PIC 'ZZZZZZ9V.99',
*====*      CF ALLOC BAL                  PIC 'ZZZZZZ9V.99',
*====*      CF UNALLOC BAL               PIC 'ZZZZZZ9V.99',
*====*      CF ORD BAL                   PIC 'ZZZZZZ9V.99',
*====*      CF CR BAL                    PIC 'ZZZZZZ9V.99';
-- END OF MEMBER --
```

EDIT session after DELETE command

```
=>
EDIT  EMX3.EMSA200                      SESS=A 1( 1) LINE=      1(      3)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*      CF UNALLOC BAL               PIC 'ZZZZZZ9V.99',
*====*      CF ORD BAL                   PIC 'ZZZZZZ9V.99',
*====*      CF CR BAL                    PIC 'ZZZZZZ9V.99';
-- END OF MEMBER --
```

DISCARD

Use DISCARD to end a LISTP or MAIL session and to delete the POWER job entry, JES job or data sets, or mail message associated with the session.

DISCARD may also be entered as DISC.

DISCARD has no operands.

DISCARD is not rejected in situations that prevent a job or message from being purged. Instead, the session is ended and a warning message may be displayed.

On VSE, POWER job entries can be purged with either DISCARD or PURGEP. DISCARD sets "last referenced POWER job entry". If purge access to a job entry is rejected, the session is terminated but the job is not purged. An 'inadequate access level' message is displayed.

On MVS, JES jobs or data sets can be purged with either DISCARD or PURGEP. DISCARD sets "last referenced JES data sets". If the session was started with the DSET operand, DISCARD will not succeed because JES provides no facility to delete by data set. DISCARD deletes all data sets within the job that met the selection for the session. If the session selection was all SYSOUT, the entire job is deleted. If purge access to a job entry is rejected, the session is terminated but the job is not purged. No message is given.

Messages can be purged with either DISCARD or PURGEQ.

See the PURGEP and PURGEQ command descriptions for more information.

Use in a Procedure

DISCARD always returns OK, even though the operation was only a partial success.

Example

End the current session, and purge the job or message associated with the session:

```
=> discard
```

DISPLAY

Use DISPLAY to create a DISP session of an existing member.

DISPLAY may also be entered as DISP or DI.

Optional Operands	
MEM	is an existing member, \$LOG, \$MAIL, or \$STACK. \$LOG is the log area, \$MAIL the mail log, and \$STACK is the stack area. If MEM is not specified, the "last referenced member" is used.
STR	is a character string that will be displayed in the information line of the DISP session. The REFRESH command will attempt to recreate a DISP session by processing the STR value as a command. STR can be up to 37 characters in length.
PSWD	If the member is password protected, the password must be entered. The password may be up to 8 characters in length consisting of any characters.

If the member is currently being edited, DISPLAY will display the text of the member ignoring any changes made in the edit session.

The DISPLAY command will create a display session identical to the LIST command if MMPDSCTL is set to zero. If MMPDSCTL is set to other than zero the session created will be updateable. When an updateable display session is terminated the changes will not be saved. This command can be useful for members containing JCL that needs to be temporarily modified prior to submission. Using this command you don't need to remember to do an END NOSAVE after an edit session and you don't need to have EDIT access security to the member.

When a DISPLAY session is created for \$LOG, \$MAIL, or \$STACK, the following member attributes will be assumed: ZONE=1-253, FCOL=1, CASE=U.

DISPLAY sets "last referenced member".

DISPLAY sets the TXM variables to the attributes of MEM.

DISPLAY can also be invoked by entering the T command in the LCA area of a LIBRARY display.

When Valid

The user must have LIST access level for the MEM library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Member not found.
SC	Inadequate access level.

Examples

Create a display session for member RM3521:

```
=> disp rm3521
```

DISPLAYC (VSE only)

Use DISPLAYC to create a display session of the VSE System Console messages.

DISPLAYC may also be entered as DISPC or DC.

Optional Operands	
FCT	<p>specifies the number of lines to be generated in the display session. Any value up to your User Size Limit can be specified. If FCT is not specified, the number of lines available in your current screen is used. (The default for a batch application is 40 lines.)</p> <p>If a value greater than the available screen size is specified for FCT, the display session will be positioned as if the BOTTOM command had been entered.</p>
PART	<p>specifies the partition for which messages are to be displayed. PART can be entered as either a 1 or 2 character value. If 1 character is entered, all partitions with the same starting character will be displayed.</p>
DATE	<p>specifies the date for the newest console message to be included in the display. If omitted, the display contains the most recent messages displayed on the System Console.</p> <p>DATE can be entered in one of the following formats: "mm/dd", "dd/mm", "mm/dd/yy", "mm/dd/yyyy", "dd/mm/yy" or "dd/mm/yyyy" depending upon the date format set by the System Administrator in field MMPDTCTL. If "mm/dd" or "dd/mm" is entered, the current year is assumed.</p>
TIME	<p>specifies the time for the newest console message to be included in the display. If omitted, the display contains the most recent messages displayed on the System Console.</p> <p>TIME can be entered as hh:mm:ss or hh:mm, and is a 24 hour time.</p> <p>If both DATE and TIME are specified, the display ends with the first message generated on or before the specified DATE and TIME.</p>
STR	<p>is a string to search for. Up to 72 characters can be specified. If the value of STR begins with the backslash character (\), the search will use Extended Search Pattern rules.</p> <p>If STR is specified in conjunction with DATE and/or TIME, the display will end with the first console message containing the specified STR that occurs on or before the specified DATE and/or TIME.</p>

Optional Operands (continued)	
ZONE	is the column range for the search. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to limit the search to columns 10 to 20, specify ZONE as "10-20". ZONE=5 is the same as ZONE=5-5, ZONE=5-* is the same as ZONE=5-253 and ZONE=-5 is the same as ZONE=1-5. If ZONE is not specified, ZONE=1-80 is assumed.
LCA	specifies whether the display should be generated with an LCA area. The default is NO. If YES is specified, an LCA will be generated on the left or right side of the display based on the setting of field SIBDFLCA.

Messages will be displayed in the session with the oldest message on top and the newest message on the bottom.

The DISPLAYC command can be interrupted during a DATE, TIME, STR or PART search by pressing the ATTN key.

The REFRESH command can be used to refresh the messages displayed in the session. If the DATE, TIME, and STR operands were not specified, the display is refreshed using the newest messages displayed on the System Console. If DATE and/or TIME were specified, and STR was not specified, the display is refreshed with the next FCT number of older messages. If STR was specified, the display is refreshed with the next FCT number of older messages, starting with the next occurrence of STR.

Each line of the session contains the following information:

- 80 character console message.
- Time message was generated in hh:mm:ss format.
- Date message was generated in mm/dd/yyyy or dd/mm/yyyy format based on the setting of field MMPDTCTL.
- Partition ID.
- Unique message number assigned by VSE/ESA 2+

The last line of the display will be one of the following messages:

```
1I87I REPLY TO: pp-nnn, ...  
1I88I NO REPLIES OUTSTANDING
```

You can invoke the MESSAGE command to display the description of an error message by using the LCA L command on the DISPLAYC line containing the message id.

Use in a Procedure

Return Codes:

- | | |
|----|--|
| OK | Successful. |
| IV | Error returned from console interface. |

Examples

Create a display session for console messages generated on or before 05/01/2000 at 11:30 am:

```
=> dc date=05/01/2000 time=11:30
```

Create a display session containing 1000 lines for console messages generated by partition BG that were created on or before 01/01/2000:

```
=> dc fct=1000 date=01/01/2000 part=bg
```

DISPLAYD (VSE version)

Use DISPLAYD to create a DISPD session of a VSE sublibrary member.

DISPLAYD may also be entered as DISPD or DID.

Optional Operands	
MEMD	the name and type of an existing VSE sublibrary member. The currently attached sublibrary (see ATTACHD) will be used unless the library and sublibrary are prefixed to the member name and type (i.e. library.sublib.member.type). If MEMD is not specified, the "last referenced VSE sublibrary member" will be used.

The initial characteristics of the DISPLAYD session are those established by the System Administrator for the member type. This is implemented by copying the attributes of a "template member" in the BIM-EDIT Site or System Model library. The template member used is the one whose name matches the member type.

DISPLAYD sets the "last referenced VSE sublibrary member".

See the HEX option of the SCREEN command for working with object members.

The DISPLAYD command will create a display session identical to the LISTD command if MMPDSCCTL is set to zero. If MMPDSCCTL is set to other than zero the session created will be updateable. When an updateable display session is terminated the changes will not be saved.

DISPLAYD can also be invoked by entering the T command in the LCA area of a LIBRARYD display.

When Valid

BIM-EDIT must be running in its own partition and under VSE/SP release 2.1 or later. The user must have LIST access level for the MEMD library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Member not found.
SC	Inadequate access level.

Examples

Create a display session of member DAVE1.PHASE in sublibrary PERSONAL.DAVE:

```
=> dispd personal.dave.dave1.phase
```

Create a display session of member JMY8806.C in the current sublibrary:

```
=> did jmy8806.c
```

Output of DISPLAYD Command

```
=>
DISPD BIMLIB.T.JMY8806.C                      SESS=A 2( 2) LINE=      0(   48)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF MEMBER --
*====*      *PROPOSED EMPLOYEE FILE RECORD STRUCTURE 6/88 JMY
*====*      01 EMPLOYEE-REC.
*====*          03 EMP-NAME.
*====*              05 EMP-FIRST-NAME      PICTURE X(20) .
*====*              05 EMP-LAST-NAME      PICTURE X(20) .
*====*          03 EMP-HIRE-DATE.
*====*              05 EMP-HIRE-MM        PICTURE 99 .
*====*              05 EMP-HIRE-DD        PICTURE 99 .
*====*              05 EMP-HIRE-YY        PICTURE 99 .
```

DISPLAYD (MVS version)

Use DISPLAYD to create a DISPD session of a Partitioned Data Set (PDS) member.

DISPLAYD may also be entered as DISPD or DID.

Optional Operands	
MEMD	the name of the PDS member. The currently attached PDS (see ATTACHD) will be used unless the PDS name is prefixed to the member name, that is, pdsname.member or pdsname(member). If MEMD is not specified, the "last referenced PDS member" will be used.

The initial characteristics of the DISPD session are those established by the System Administrator for the member type. This is implemented by copying the attributes of a "template member" in the BIM-EDIT Site or System Model library. The template member used is the one whose name matches the last qualifier in the PDS name.

DISPLAYD sets the "last referenced PDS member".

DISPLAYD is allowed only for fixed block members with a record length 253 or less. Members containing source code generally meet this qualification, but load modules do not.

The DISPLAYD command will create a display session identical to the LISTD command if MMPDCTL is set to zero. If MMPDCTL is set to other than zero the session created will be updateable. When an updateable display session is terminated the changes will not be saved.

DISPLAYD can also be invoked by entering the T command in the LCA area of a LIBRARYD display.

When Valid

The user must have LIST access level for the MEMD library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Member not found.
SC	Inadequate access level.

Examples

Create a display session of member DAVE1 in PDS PERSONAL.DAVE:

```
=> dispd personal.dave.dave1
```

Create a list session of member JMY8806 in the current sublibrary:

```
=> did jmy8806
```

Output of DISPLAYD Command

```
=>
DISPD BIMLIB.JMY8806                      SESS=A 2( 2) LINE=      0(   48)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*  -- TOP OF MEMBER --
*====*          *PROPOSED EMPLOYEE FILE RECORD STRUCTURE 6/88 JMY
*====*          01  EMPLOYEE-REC.
*====*              03  EMP-NAME.
*====*                  05  EMP-FIRST-NAME          PICTURE X(20) .
*====*                  05  EMP-LAST-NAME           PICTURE X(20) .
*====*              03  EMP-HIRE-DATE.
*====*                  05  EMP-HIRE-MM              PICTURE 99 .
*====*                  05  EMP-HIRE-DD              PICTURE 99 .
*====*                  05  EMP-HIRE-YY              PICTURE 99 .
```

DISPLAYS

Use DISPLAYS to create a display session containing a line for each active session .

DISPLAYS may also be entered as DISPS or DS.

DISPLAYS has no operands.

The LCA 'S' command can be used to select a desired session. The DISPLAYS session will be terminated and the appropriate GROUP and SESSION commands will be generated internally to cause the selected session to become the active session.

Use in a Procedure

Return Codes:

OK Successful.

Example

Create a display session of all active sessions.

```
=> ds
```

```
=>
DISP  -> ds                                SESS=B 4 ( 4) LINE=    0 (   11)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----
*====* -- TOP OF DISPLAY --
*====* SESS DESCRIPTION
*====* -----
*====* A1  DISP  -> lib
*====* A2  LIST  SJA.TEXT
*====* A3  LIST  SJA.PROC
*====* A4  LIST  SJA.JCL
*====* A5  LIST  SJA.EDITMEM
*====* B1  DISP  -> lib $sys.ctrl.
*====* B2  LIST  $SYS.CTRL.MESSAGES
*====* B3  LIST  $SYS.CTRL.LOGON
*====* B4  DISP  -> ds
-- END OF DISPLAY --
```

DOWN

Use DOWN to position the current session forward (toward the bottom) a specified number of lines.

DOWN may also be entered as N, NEXT, or DO.

Optional Operands	
FCT	specifies the number of lines to move the position by. If FCT is not specified, BIM-EDIT will position forward one line.

Use in a Procedure

Return Codes:

- OK Successful
- EF Already positioned at the last line of the session (this only occurs when NEXT is executed from a procedure).

Examples

Position forward 29 lines:

```
=> down 29
```

Position forward 1 line:

```
=> do
```


EDIT

Use EDIT to create an EDIT session of an existing BIM-EDIT member.

EDIT may also be entered as ED.

Optional Operands	
MEM	is the member to be edited. If MEM is not specified, the "last referenced member" is used.
PSWD	If the member is password protected, the password must be entered before edit access is allowed. The password may be up to 8 characters in length consisting of any character.

EDIT can also be invoked by entering the LCA E command in the LCA area of a LIBRARY display.

Before MEM can be edited, it must be created by either the DEFINE or COPY commands.

EDIT creates a session with the text and attributes of an existing member. MEM is updated only when the EDIT session is saved (see the SAVE command). Before the session is saved, you can use an END NOSAVE command to discard the EDIT session without updating the member.

The following LCA commands can be useful in conjunction with the EDIT session:

A	Add lines	(LADD)
B	Insert lines from \$STACK before	(GET \$STACK)
C	Copy lines to \$STACK	(STACK)
D	Delete lines	(DELETE)
I	Insert lines from \$STACK after	(GET \$STACK)
J	Merge lines from \$STACK	(MERGE)
K	Copy additional lines to \$STACK	(STACK)
M	Move lines to \$STACK	(STACK;DELETE)
N	Move additional lines to \$STACK	(STACK;DELETE)
U	Translate lines to uppercase	(UPPERCAS)
W	Translate lines to lowercase	(LOWERCAS)
"	Duplicate lines	(DUP)
/	Position display	(POSITION)
<	Shift lines left	(SHIFT)
>	Shift lines right	(SHIFT)
+	Center lines	(CENTER)
(Justify lines left	(JUSTIFYL)
)	Justify lines right	(JUSTIFYR)

See Chapter 5, LCA Commands, in the BIM-EDIT User Reference Manual for more information.

EDIT sets the "last referenced member".

When Valid

The user must have EDIT access level for the MEM library. MEM cannot be currently undergoing editing. MEM cannot be checkout slave assigned to another user. MEM cannot be a checkout master (have CHECK=ON) -- use the CHECKOUT command to create a slave (working) copy.

Use in a Procedure

Return Codes:

OK	Successful.
CK	One of the following: <ul style="list-style-type: none">• Member is under checkout control.• Member is checked out to another user.
ED	Member is already being edited.
NF	Member not found.
SC	Inadequate access level.

EDIT sets the TXM variables to the attributes of MEM.

Example

Edit member RM3521:

```
=> edit rm3521
```

EDITD (VSE version)

Use EDITD to create an EDIT session of a VSE sublibrary member.

EDITD may also be entered as EDD.

Optional Operands	
MEMD	the name and type of an existing VSE sublibrary member. The currently attached sublibrary (see ATTACHD) will be used unless the library and sublibrary are prefixed to the member name and type (i.e. library.sublib.member.type). If MEMD is not specified, the "last referenced VSE sublibrary member" will be used.

The edit session created by this command behaves just as described in the EDIT command.

The initial characteristics of the EDITD session are those established by the System Administrator for the member type. This is implemented by copying the attributes of a "template member" in the BIM-EDIT Site or System Model library. The template member used is the one whose name matches the member type.

EDITD sets the "last referenced VSE sublibrary member".

See the HEX mode of the SCREEN command for working with object members. "PHASE" type members cannot be edited.

When Valid

BIM-EDIT must be running in its own partition and under VSE/SP release 2.1 or later. The user must have EDIT access level for the MEMD library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Member not found.
SC	Inadequate access level.

Examples

Edit member CJO1723.OBJ (a relocatable member) in the current sublibrary:

```
=> editd cjo1723.obj
```

Edit the same member in the BIMLIB.ED51A sublibrary:

```
=> edd bimlib.ed51a.cjo1723.obj
```

EDITD (MVS version)

Use EDITD to create an EDIT session of a Partitioned Data Set member.

EDITD may also be entered as EDD.

Optional Operands	
MEMD	the name of the PDS member. The currently attached PDS (see ATTACHD) will be used unless the PDS name is prefixed to the member name, that is, pdsname.member or pdsname(member). If MEMD is not specified, the "last referenced PDS member" will be used.

The edit session created by this command behaves just as described in the EDIT command.

The initial characteristics of the EDITD session are those established by the System Administrator for the member type. This is implemented by copying the attributes of a "template member" in the BIM-EDIT Site or System Model library. The template member used is the one whose name matches the last qualifier in the PDS name.

EDITD sets the "last referenced PDS member".

EDITD is allowed only for fixed block members with a record length 253 or less. Members containing source code generally meet this qualification, but load modules do not.

When Valid

The user must have EDIT access level for the MEMD library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Member not found.
SC	Inadequate access level.

Examples

Edit member CJO1723 in the current PDS:

```
=> editd cjo1723
```

Edit the same member in the BIM0001.TESTLIB.D880623 PDS:

```
=> edd bim0001.testlib.d880623.cjo1723
```

Edit member prog428 in SYSLIB.PROBLEM PDS:

```
=> editd syslib.problem(prog428)
```

END

Use END to terminate the current session.

Optional Operands	
OPT	can be specified as "NOSAVE" to terminate an EDIT or EDITD session without updating the associated member. BIM-EDIT allows an EDIT or EDITD session to be terminated with the END command only if no updates have taken place or OPT is specified as "NOSAVE". Use the SAVE command to terminate an EDIT or EDITD session and update the member.

After the session is ended, the screen will display the previous session in the session chain.

When an EDIT or LIST session is ended, the "last referenced member" entry (variables TXMLIB, TXMMEM and TXMPATH) are set to the member for the session ended.

When a LISTP session is ended, the "last referenced POWER job entry" (variables PWRJNAME, PWRJNUM and PWRTYPE) or "last referenced JES data sets" (all of the JQE variables) is set to the entry for the session ended.

When an EDITD or LISTD session is ended, the "last referenced VSE sub- library member" or "last referenced MVS PDS member" (variable SIBLSPDS) is set to the entry for the session ended.

This scheme for setting "last referenced ..." is useful because there are often further commands desired (such as PROCESS or PURGEP) against the entity ended. It is also confusing because "last referenced" does not correspond to what is showing on the screen after an END.

Use in a Procedure

END normally returns OK. (See SIBRETCDD.)

Examples

End the current DISP, LIST, LISTD, LISTP, or MAIL session:

```
=> end
```

End the current EDIT session (no updates have occurred):

```
=> end
```

End the current EDIT session without saving text (updates have occurred):

```
=> end nosave
```

EXECUTE

Use EXECUTE to process the text of a member as commands (i.e. as a procedure).

EXECUTE may also be entered as EXEC or EX.

Optional Operands	
MEM	is the member to be executed. If MEM is not specified, the "last referenced member" is used. The member to be executed must be of type "PROC". MEM can also be "\$STACK", in which case the contents of the stack area are executed as a procedure or "/", in which case the current session is executed. To execute a REXX procedure, the member type must be "REXX".
.....	Additional parameters are placed in the PARMLIST variable. These parameters are retrievable from within the procedure via the PARSE command.

EXECUTE is often invoked through the PROCESS command, which in turn is invoked from either the command line or the LCA S command. See the PROCESS command documentation for more information.

If MEM is not the same as a command name, the procedure can be executed simply by entering MEM on the command line without the EXECUTE. For example, if a procedure called "SMAP" existed, it could be executed by the following:

```
=> smap
```

On the other hand, a procedure of name "NEXT" could not be executed in this manner since "NEXT" is the name of a command. This "implied" form of procedure execution is limited to procedures in the currently attached library whose names are 8 characters or less in length.

A procedure can be made into a command by specifying the procedure in a command table entry. For more information, see Chapter 11, Customization, in the BIM-EDIT System Reference Manual.

When using EXECUTE / to execute the current session, commands which would change the current session (such as END, ROTATE, EDIT) are not allowed.

Procedures can be extremely useful where repetitive tasks are performed. By creating a procedure and executing it, the procedure processor will pass to the command processor, one by one, the lines of a member. The result is exactly the same as if the commands had been entered through the command line.

Procedures can themselves execute procedures. The maximum nesting level permitted is 4.

For more extensive discussion and examples, see Chapter 2, Writing Procedures, in the BIM-EDIT System Reference Manual.

EXECUTE does not set "last referenced member" (but the procedure executed may).

When Valid

The user must have EXEC access level for the MEM library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Procedure not found.
SC	Inadequate access level.
*	Other codes as determined by commands within the procedure itself.

Example

Execute the procedure SMAP, passing two parameters to the procedure:

```
=> execute smap rmgl030,xref=yes
```

or

```
=> smap rmgl030,xref=yes
```

FALTER

Use FALTER to display and alter the attributes of a member.

FALTER may also be entered as FALT.

Optional Operands	
MEM	is the member to be altered. If MEM is not specified, the "last referenced member" is used.

FALTER can also be invoked by entering the Q command in the LCA area of a LIBRARY display.

FALTER creates a display showing all member attributes. You can change an attribute simply by overtyping its current value.

When an EDIT or LIST session of a member is created, the member's attributes become the session's attributes. Altering a member's attributes after an EDIT or LIST session of the member is created has no effect on that session. If you want to affect the attributes of an existing session, use the SESS command.

FALTER sets "last referenced member".

When Valid

FALTER can only be used as an online command or in an online procedure. The user must have DEF access level for the MEM library.

Use in a Procedure

Return Codes:

OK	Successful.
CK	Member is part of a checkout relationship.
NF	Member not found.
SC	Inadequate access level.

FALTER sets the TXM variables to the attributes of MEM.

FALTER is implemented as the system procedure BIPFALM.

Example

Alter member BIPSTKY:

```
=> falter bipstky
```


Screen displayed by FALTER

FALTER			
MEMBER	: ED54A.BIPSTKY		

TITLE	: SET PF KEYS		
TYPE	: PROC		
ATTR	: 5.6A	CHECKOUT USER	:
LIBDEF	:	CHECKOUT DATE/TIM	:
USER	: PG	CHECKOUT LIBRARY	:
CASE	: U	DATE/TIME CREATED	: 09/09/1999 19:57:52
NULLS	: ON	DATE/TIME UPDATED	: 00/00/0000 00:00:00
ZONE	: 1-128	CREATED BY	: PG
SEQ NUMBER COL	:	LAST UPDATED BY	:
FIND COLUMN	: 1	ED SESSION USER	:
WORD WRAP (/XP)	: OFF	LIST SESSION USER	:
AUDIT STATUS	: ON	SIZE	: 25
STAMP STATUS	: OFF	AUDIT SIZE	: 0
CHECKOUT STATUS	: OFF		
TAB COLUMNS	:		
CURRENT PASSWORD	:	VERIFY PASSWORD	: Password or 'NO'
NEW PASSWORD	:		
(PF1/PF13 = Update and End. PF2/PF14 Update and Redisplay. CLEAR/PF3 = End.)			

The attributes displayed on this screen may be changed by overtyping them.

FALTERL

Use FALTERL to display and alter the attributes of a library.

FALTERL may also be entered as FALTL.

Required Operands	
LIB	is the library to be altered.

FALTERL can also be invoked by entering the Q command in the LCA area of a LIBRARYL display.

FALTERL creates a display showing all attributes for the library. You can change an attribute simply by overtyping its current value.

When Valid

FALTERL can only be used as an online command or in an online procedure. The user must have DEFL access level for the library prefix of the LIB library (the part of the library name preceding an imbedded period). If LIB has no prefix, the user must have system-wide DEFL access.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Library not found.
SC	Inadequate access level.

FALTERL is implemented as the system procedure BIPFALL.

Examples

Alter library DFH:

```
=> falterl dfh
```

Screen displayed by FALTERL

```
FALTERL
LIBRARY      : DFH

-----
TITLE        : CICS TABLES (PCT, FCT, ETC.)
USER         : $SYS

(PF1, PF13 = Update and End. PF2, PF14 Update and Redisplay. CLEAR=End.)
```

The attributes displayed on this screen may be changed by overtyping them.

FALTERP (VSE version)

Use FALTERP to display and alter the attributes of a POWER job entry.

FALTERP may also be entered as FALTP.

Optional Operands	
PRM1-4	<p>select the POWER job entry to be altered. The four operands allow you to specify the queue (RDR,LST,PUN,XMT), the job name, the job number, and the segment number. They can be specified in any order, except that segment, if specified, must follow job number. The following indicates which entry will be selected based on the operands specified:</p> <p>queue, job name, job number, segment</p> <p>The specified queue will be searched for the entry for the specified name, number, and segment.</p> <p>queue, job name</p> <p>The specified queue will be searched for the first entry for the specified name.</p> <p>queue, job number</p> <p>The specified queue will be searched for the first entry for the specified number, without regard to the job name.</p> <p>job number</p> <p>job name</p> <p>The LST queue will be searched for the first entry for the specified number or name.</p> <p>queue</p> <p>The specified queue will be searched for the first entry for the job name and number of the "last referenced POWER job entry"</p> <p>If PRM1-4 is not specified at all, the "last referenced POWER job entry" will be used.</p>

FALTERP can also be invoked by entering the Q command in the LCA area of a LIBRARYYP display.

FALTERP creates a display showing all POWER job entry attributes. You can change an attribute simply by overtyping its current value.

FALTERP sets the "last referenced POWER job entry".

When Valid

FALTERP can only be used as an online command or in an online procedure. Check with your System Administrator to determine the restrictions on access to POWER job

entries at your site. The exit routine distributed with BIM-EDIT provides several restriction schemes your System Administrator can select among. (Use of and changes to the BIXPWQA exit routine are described in the System Reference Manual.)

Use in a Procedure

Return Codes:

OK	Successful.
NF	Job not found.
PW	Rejected by POWER.
*	Other codes as set by BIXPWQA exit routine.

FALTERP sets the PWR variables to the attributes of the POWER job entry.

FALTERP is implemented as the system procedure BIPFALP.

Examples

Alter a job entry in LST queue, identifying the job with queue and job number.

```
=> falterp lst,1628
```

Screen displayed by FALTERP

```
FALTERP
POWER JOB      : LST,LL01$BIM,1628
-----
CLASS          : A
DISPOSITION    : D
PRIORITY       : 3
COPIES         : 1
SYSTEM ID      :
ORIGIN NODE    : LOCAL
ORIGIN USER    : R000
DESTINATION NODE : LOCAL
DESTINATION USER : LOCAL
FORMS ID       :
USER INFO      : BM01
DATE           : 06/15/95
PAGES          : 2
RECORDS DATA/CTRL : 3206 / 3283
DATA BLOCK GROUPS : 13
NEXT RUN DUE AT : 14:00 ON 2000/12/08
(PF1, PF13 = Update and End. PF2, PF14 Update and Redisplay. CLEAR=End.)
```

The attributes displayed on this screen may be changed by overtyping them.

FALTERP (MVS version)

Use FALTERP to display and alter the attributes of a JES job or data sets.

FALTERP may also be entered as FALTP.

Optional Operands	
JOB	selects the JES job to be altered. If JOB contains letters, it specifies a jobname label from the Job Control Language JOB statement. If JOB is entirely digits, it specifies a job number assigned by JES. If not specified, the job from "last referenced JES data sets" will be used.
GROUP	<p>selects the data sets to be altered within the specified job. To alter a JES job (as opposed to data sets), do not specify GROUP.</p> <p>If GROUP is a single letter or digit, it specifies all data sets in the GROUP class in the JES "hold queue". If GROUP is any other multiple character value, it specifies all data sets in the GROUP outgrp in the JES "output queue".</p> <p>An outgrp may be composed of three parts separated by periods in the form "name.qualifier.copy" where "name" is the JES or user-specified name, qualifier is a JES-assigned number, and copy is the copy number. All data sets matching a partial outgrp specification (for example, "name" only) will be altered. To distinguish an outgrp with a one-character name from a class, follow the name with a period.</p> <p>If neither JOB or GROUP is specified, the "last referenced data sets" are altered. (If "last referenced data sets" was specified with a DSET operand, FALTERP will be rejected because JES provides no facility to alter by data set.)</p>

FALTERP can also be invoked by entering the Q command in the LCA area of a LIBRARYP display.

FALTERP creates a display showing JES job or data set attributes. You can change an attribute simply by overtyping its current value.

FALTERP sets "last referenced JES data sets".

FALTERP alters all data sets within the job that meet the selection.

When Valid

FALTERP can only be used as an online command or in an online procedure. Check with your System Administrator to determine the restrictions on access to JES data sets at your site. The exit routine distributed with BIM-EDIT provides several restriction schemes your System Administrator can select among. (Use of and changes to the BIXPWQA exit routine are described in the System Reference Manual.)

Use in a Procedure

Return Codes:

OK	Successful.
NF	JES job not found.
*	Other codes as set by BIXPWQA exit routine.

FALTERP sets the JCT, JQE, and PDB variables to the attributes of the JES data sets altered. If GROUP is an outgrp, the JOE variables are also set.

FALTERP is implemented as the system procedure BIPFALP.

Examples

Alter hold queue class A data sets for job 1934

```
=> falterp 1934,a
```

Alter job 1628

```
=> faltp 1628
```

Screen displayed by FALTERP

```
FALTERP
JES JOB          : BICEDTD/8083
-----
CLASS            : I
PRIORITY         : 7
TYPE             : J
MESSAGE CLASS    : A
REMOTE NODE      : 1
INPUT/OUTPUT LINES : 20 / 150
ROOM # / PROGRAMMER : /
ACCOUNT # / NOTIFY : ANX218 / RYU482
DATE/T READ IN   : 08/11/95 13:01:12
DATE/T EXEC BEGAN : 08/11/95 13:02:40
INPUT/EXEC SYSTEMS : A / A

(PF1, PF13 = Update and End. PF2, PF14 Update and Redisplay. CLEAR=End.)
```

The CLASS and PRIORITY attributes displayed on this screen may be changed by overtyping them.

FCOPY

Use FCOPY to create a copy of an existing member.

Required Operands	
SRC	is an already existing member.
DEST	is the name of the member to be created.

Optional Operands	
PSWD	If the member is password protected, the password must be entered. The password may be up to 8 characters in length consisting of any characters. A password protected member that is copied will have the same password as the original member.
REPL	is used to specify whether the DEST member is to be replaced if a member by this name already exists. The default is "NO". If REPL=YES is specified and the DEST member does not exist the FCOPY will be completed but the completion message will display the fact that the DEST member was not found. It is not possible to replace the following: <ul style="list-style-type: none"> • a master member (i.e. check status is on) • a member that is part of a checkout relationship • a member that is password protected

FCOPY creates a display showing the required and optional operands that may be used to copy one member to a new member name.

A copy of the source member will be created under the destination name. FCOPY, COPY and DEFINE are the only commands that create new members. Both the text and the member attributes are copied. The source member is unaffected by the FCOPY.

If an asterisk (*) is entered as part of DEST, the asterisk is replaced with the corresponding part of SRC member name. This facilitates copying a member from one library to another or making a copy of a member to a suffixed or prefixed name without keying the member name twice (see Examples).

To move a member from one library to another, see the MOVE, RENAME or RENAME commands.

FCOPY sets "last referenced member" to the DEST member.

When Valid

FCOPY can only be used as an online command or in an online procedure. The user must have LIST access level for the SRC library and DEF access level for the DEST library.

Use in a Procedure

Return Codes:

OK	Successful.
DP	DEST member already exists.
NF	Either the SRC member was not found or the DEST library was not found.
SC	Inadequate access level.

FCOPY sets the TXM variables to the attributes of DEST.

Examples

Request full screen COPY prompt:

```
=> fcopy
```

Screen displayed by FCOPY

```
FCOPY
SOURCE MEMBER      :                      (required)
DESTINATION MEMBER :                      (required)
-----
PASSWORD           :
REPLACE OPTION     : NO
(PF1/PF13 = Update and End. PF2/PF14 Update and Redisplay. CLEAR/PF3 = End.)
```


FDEFINE

Use FDEFINE to create a new member by filling in attributes on the screen.

FDEFINE may also be entered as FDEF.

Optional Operands	
MEM	<p>is the member to be defined. MEM can be up to 16 characters in length, composed of the letters A-Z, the numerals 0-9, and the special characters dash (-), dollar sign (\$), pound sign (#), and the underscore (_). (The libname.memname format can also be used. In that case, MEM is up to 33 characters in length: 16 for libname, 1 for period, 16 for memname.)</p> <p>While BIM-EDIT supports the underscore (_) in the member name, it is not supported by all of the subsystems that BIM-EDIT interfaces with (for example, POWER). Care should be used when deciding which member names may contain underscores.</p> <p>If MEM is omitted, the full screen display will prompt the user for the required and optional operands.</p>
TYPE	<p>is the member type, which determines the initial attributes and contents of the new member. Member attributes from TYPE can be overridden by using other operands on the DEFINE. If TYPE is enclosed in parentheses (), the member will not have any initial contents. TYPE can be up to 8 characters in length. If TYPE is not specified, the type defined for model \$DFL is used.</p> <p>For program source members, TYPE is also typically used by the COMPILE command to determine which compiler to create Job Control Language for.</p> <p>BIM-EDIT, as distributed, provides the following TYPEs:</p> <p>\$DFL Default template. ASM Assembler program source. COBOL COBOL program source. DATA Data. FORT FORTRAN program source. JCL DOS job streams. PLI PLI program source. PROC Procedure. RPG RPG program source. TEXT Textual material, such as documentation.</p> <p>Your System Administrator has probably added or deleted TYPEs from the above list.</p> <p>TYPE is implemented by copying the attributes and text of a "template member" of name TYPE in the BIM-EDIT Site or System Model library to the new member.</p>

FDEFINE creates a display showing the default member attributes. You can change an attribute by simply overtyping its current value. See the DEFINE command for discussion of specific attributes.

FDEFINE sets "last referenced member".

When Valid

FDEFINE can only be used as an online command or in an online procedure. The user must have DEF access level for the MEM library.

Use in a Procedure

Return Codes:

OK	Successful.
DP	Member already exists.
NF	One of the following: <ul style="list-style-type: none">Library not found.TYPE is not valid.
SC	Inadequate access level.

FDEFINE sets the TXM variables to the attributes of MEM.

FDEFINE is implemented as the system procedure BIPFDFM.

Examples

Define member 361-100 in library IS as a PL/1 member:

```
=> fdefine is.361-100,pli
```

Screen displayed by FDEFINE

```
FDEFINE
MEMBER      : IS.361-100                                (required)
-----
TITLE       : PROJECT COST STATUS DETAIL
TYPE        : PLI
ATTRIBUTES  :
LIBDEF      :
USER        : GOP
CASE        : U
NULLS       : ON
ZONE        : 1-72
SEQ NUMBER COL : 73-80
FIND COLUMN : 1
WORD WRAP (/XP) : OFF
AUDIT STATUS : OFF
STAMP STATUS : OFF
CHECKOUT STATUS : OFF
TAB COLUMNS :
PASSWORD    :          VERIFY PASSWORD:

(PF1/PF13 = Update and End. PF2/PF14 Update and Redisplay. CLEAR/PF3 = End.)
```

The attributes displayed on this screen may be changed by overtyping them. For more discussion about a particular attribute, see the DEFINE command.

FDEFINEL

Use FDEFINEL to create a new library by filling in attributes on the screen.

FDEFINEL may also be entered as FDEFL.

Optional Operands	
LIB	is the library to be created. LIB can be up to 16 characters in length, composed of the letters A-Z, the numerals 0-9, and the special characters period (.), dash (-), dollar sign (\$), pound sign (#), and the underscore (_). If LIB is omitted, the full screen display will prompt the user for the required and optional operands.

FDEFINEL creates a display showing the default library attributes. You can change an attribute by simply overtyping its current value. See the DEFINEL command for discussion of specific attributes.

When Valid

The user must have DEFL access level for the library prefix of the LIB library (the part of the library name preceding an imbedded period). If LIB has no prefix, the user must have system-wide DEFL access.

Use in a Procedure

Return Codes:

OK	Successful.
DP	Library already exists.
SC	Inadequate access level.

FDEFINEL sets the TXL variables to the attributes of the LIB library. See Chapter 4, Predefined Variables, in the BIM-EDIT System Reference Manual.

FDEFINEL is implemented as the system procedure BIPFDFL.

Examples

Define library 4216:

```
=> fdefin1
```

Screen displayed by FDEFINEL

```

FDEFINEL
LIBRARY      : 4216
-----
TITLE        : PROJECT COST STATUS DETAIL
USER         : MEA
(PF1, PF13 = Update and End. PF2, PF14 Update and Redisplay. CLEAR=End.)

```

The attributes displayed on this screen may be changed by overtyping them. For more discussion about a particular attribute, see the DEFINEL command.

FIND

Use FIND to position the current session forward from the current line to the first line which contains a specified string of characters at a specified column.

FIND may also be entered as F.

Optional Operands	
STR	is the string to search for. Up to 72 characters may be specified. If the value of STR begins with the backslash character (\), the search will use Extended Search Pattern rules, as described below. If STR is not specified, the value for STR and any other operands not specified will be inherited from previous search commands, also as described below.
COL	specifies the column to search at. COL must be a value from 1 to 253. If COL and STR are not specified, the COL specification is inherited from previous search commands. If COL, after any inheriting, is not specified, the current session FCOL value is used (see the DEFINE, ALTER, and SESS commands).
CASE	specifies whether upper/lower case should be considered when determining matches. If specified as "U", case is ignored ("the" matches "The"). If specified as "M", both case and letters must match. If CASE and STR are not specified, the CASE specification is inherited from previous search commands, as described below. If CASE, after any inheriting, is not specified, "U" is assumed.

The search begins with the line following the current line and proceeds until a match is found. An error message will display if no match is found.

After FIND, the CURSOR command will place the cursor at the match.

If your terminal is running under control of SNA, you may interrupt a long FIND command by pressing the ATTN key.

If the value of STR begins with a backslash (\), BIM-EDIT treats STR as an Extended Search Pattern, with certain characters having special meanings:

- ? matches any single character
- * matches any sequence of zero or more characters
- @ matches zero or more instances of the character that follows the @
- ! illogical - beginning of zone is implied for a FIND request
- < matches before the COL column or any non-alphanumeric character
- > matches the end of the line or any non-alphanumeric character
- | separates two patterns, EITHER of which may match for a successful search
- + illogical - FIND only searches one column for an argument

- ~ (NOT) reverses line-by-line success/failure of the pattern following it
- % marks the position on the line where the match is considered to occur
- \ suppresses the special meaning of the character that follows the \

See Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual for more information about Extended Search Patterns.

If STR is not specified on the command, BIM-EDIT uses as default operand values those from previous search commands. Using this feature, you can easily request searches which are similar or identical to the last one. For example, after a FIND command which failed because the instance is backward from the current line, you need only enter

```
=> findup
```

to repeat exactly the same command in the backward direction. To achieve this, BIM-EDIT stores a previous value for the STR, COL, CASE, and ZONE operands of the search commands (EXAMINE, FIND, FINDUP, LOCATE, LOCATEU, NFIND, NFINDUP, QUALIFY, or SCAN). These previous values will be inherited by operands omitted from a search command which does not specify STR. Whenever a search command specifies STR, the previous values are set to those specified on that command or "not specified".

Use in a Procedure

Return Codes:

- OK Successful.
- AT Command interrupted by ATTN key.
- NF String not found.

FIND sets the predefined variable SSDCOL1 to the column where the match was found if the search is successful (if in split-screen mode, and if the lower logical screen is primary, SSDCOL2 is set instead).

Examples

Search for COBOL label "A20-READ" (assume that the session FCOL is 8):

```
=> find a20-read
```

Search for the string "Examples" in column 1 (case dependent search):

```
=> f "Examples",1,case=m
```

Search for the same string as the last search:

```
=> f
```

Search for "DC" or "DS" in column 10:

```
=> find \DC|DS 10
```

FINDUP

Use FINDUP to position the current session backward from the current line to the first line which contains a specified string of characters at a specified column.

FINDUP may also be entered as FINDU, FUP, or FU.

Optional Operands	
STR	is the string to search for. Up to 72 characters may be specified. If the value of STR begins with the backslash character (\), the search will use Extended Search Pattern rules, as described below. If STR is not specified, the value for STR and any other operands not specified will be inherited from previous search commands, also as described below.
COL	specifies the column to search at. COL must be a value from 1 to 253. If COL and STR are not specified, the COL specification is inherited from previous search commands. If COL, after any inheriting, is not specified, the current session FCOL value is used (see the DEFINE, ALTER, and SESS commands).
CASE	specifies whether upper/lower case should be considered when determining matches. If specified as "U", case is ignored ("the" matches "The"). If specified as "M", both case and letters must match. If CASE and STR are not specified, the CASE specification is inherited from previous search commands, as described below. If CASE, after any inheriting, is not specified, "U" is assumed.

The search begins with the line preceding the current line and proceeds upward until a match is found. An error message will display if no match is found.

After FINDUP, the CURSOR command will place the cursor at the match.

If your terminal is running under control of SNA, you may interrupt a long FINDUP command by pressing the ATTN key.

If the value of STR begins with a backslash (\), BIM-EDIT treats STR as an Extended Search Pattern, with certain characters having special meanings:

- ? matches any single character
- * matches any sequence of zero or more characters
- @ matches zero or more instances of the character that follows the @
- ! matches before the COL column
- < matches before the COL column or any non-alphanumeric character
- > matches the end of the line or any non-alphanumeric character
- | separates two patterns, EITHER of which may match for a successful search
- + separates two patterns, BOTH of which must match for a successful search

- ~ (NOT) reverses line-by-line success/failure of the pattern following it
- % marks the position on the line where the match is considered to occur
- \ suppresses the special meaning of the character that follows the \

See Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual for more information about Extended Search Patterns.

If STR is not specified on the command, BIM-EDIT uses as default operand values those from previous search commands. Using this feature, you can easily request searches which are similar or identical to the last one. For example, after a FINDUP command which failed because the instance is forward from the current line, you need only enter

```
=> find
```

to repeat exactly the same command in the forward direction. To achieve this, BIM-EDIT stores a previous value for the STR, COL, CASE, and ZONE operands of the search commands (EXAMINE, FIND, FINDUP, LOCATE, LOCATEU, NFIND, NFINDUP, QUALIFY, or SCAN). These previous values will be inherited by operands omitted from a search command which does not specify STR. Whenever a search command specifies STR, the previous values are set to those specified on that command or "not specified".

Since it is not possible to position AFTER the last line in a session, any match for STR on the last line will be ignored by FINDUP.

Use in a Procedure

Return Codes:

- | | |
|----|----------------------------------|
| OK | Successful. |
| AT | Command interrupted by ATTN key. |
| NF | String not found. |

FINDUP sets the predefined variable SSDCOL1 to the column where the match was found if the search is successful (if in split-screen mode, and if the lower logical screen is primary, SSDCOL2 is set instead).

Examples

Search backward for COBOL label "A20-READ" (assume that session FCOL is 8):

```
=> findup a20-read
```

Search backward for string "Examples" in column 1 (case dependent search):

```
=> fu "Examples",1,case=m
```

Search backward for the same string as the last search:

```
=> fu
```

Search backward for "DC" or "DS" in column 10:

```
=> findup \DC|DS 10
```

FORMAT

Use FORMAT to combine current session text lines starting with the current line to form a paragraph in a specified column range.

FORMAT may also be entered as FORM.

Optional Operands	
ZONE	is the column range within which to format the text. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to format text with columns 14 to 77, specify ZONE as "14-77". ZONE=5 is the same as ZONE=5-5, ZONE=5-* is the same as ZONE=5-253 and ZONE=-5 is the same as ZONE=1-5. If ZONE is not specified, text is formatted within the current session zone (see the ZONE operand of the DEFINE, ALTER, and SESS commands.)
ALIGN	Specify "ON" to align text on both margins. Specify "OFF" to align text on only the left margin, that is, to leave a "ragged" right margin. If ALIGN is not specified, text will be aligned on both margins.

FORMAT updates a group of lines such that each line is filled to its maximum with text. A new line is begun only when the next word cannot fit on the line currently being processed. If the ALIGN option is on, FORMAT will intersperse extra spaces in the line so as to align text on both margins.

Formatting begins with the current line and proceeds until an end-of-paragraph indication is found. End of paragraph is indicated by a line (after the current line) with a blank, a period (.), an asterisk (*), a colon (:), or a dash (-) in the first position of the zone.

Normally, FORMAT will ignore extraneous blanks (more than one blank) when processing text. However, FORMAT will retain blanks that serve to indent the first line of a paragraph.

FORMAT updates text only in the ZONE, with one important exception: if the formatted text occupies fewer lines than the original paragraph, the extra lines are deleted, without considering whether they contain any text outside the ZONE. Since FORMAT does not alter text outside the ZONE, FORMAT can be used against text held in tables or lists. For example, the text to the right of "ZONE" in the operand table above is formatted with the following command:

```
=> format 14-77
```

FORMAT is often used in conjunction with SEPARATE. SEPARATE updates a group of lines such that each phrase begins on a new line.

Use in a Procedure

FORMAT normally returns OK. (See SIBRETCO.)

Examples

The following examples assume the current session ZONE is 1-79.

EDIT session before FORMAT command (ALIGN=ON)

```
=> format
EDIT 3217.WORK                                SESS=A 1( 1) LINE= 4442( 11251)
----|----1----|----2----|----3----|----4----|----5----|----6----|----7----|----
FORMAT is often used in conjunction with SEPARATE. SEPARATE updates a group of
SEPARATE updates a group of text lines composing a
paragraph such that each phrase begins on a new line.
Editing a paragraph is much easier after SEPARATE.
When editing is complete,
the paragraph is formatted with FORMAT.
```

EDIT session after FORMAT command (ALIGN=ON)

```
=>
EDIT 3217.WORK                                SESS=A 1( 1) LINE= 4442( 11249)
----|----1----|----2----|----3----|----4----|----5----|----6----|----7----|----
FORMAT is often used in conjunction with SEPARATE. SEPARATE updates a group of
text lines composing a paragraph such that each phrase begins on a new line.
Editing a paragraph is much easier after SEPARATE. When editing is complete,
the paragraph is formatted with FORMAT.
```

EDIT session before FORMAT command (ALIGN=OFF)

```
=> format ,off
EDIT 3217.WORK                                SESS=A 1( 1) LINE= 4442( 11251)
----|----1----|----2----|----3----|----4----|----5----|----6----|----7----|----
FORMAT is often used in conjunction with SEPARATE. SEPARATE updates a group of
SEPARATE updates a group of text lines composing a
paragraph such that each phrase begins on a new line.
Editing a paragraph is much easier after SEPARATE.
When editing is complete,
the paragraph is formatted with FORMAT.
```

EDIT session after FORMAT command (ALIGN=OFF)

```
=>
EDIT 3217.WORK                                SESS=A 1( 1) LINE= 4442( 11249)
----|----1----|----2----|----3----|----4----|----5----|----6----|----7----|----
FORMAT is often used in conjunction with SEPARATE. SEPARATE updates a group of
text lines composing a paragraph such that each phrase begins on a new line.
Editing a paragraph is much easier after SEPARATE. When editing is complete,
the paragraph is formatted with FORMAT.
```

FORWARD

Use FORWARD to position the current session down (toward the bottom) by one or more logical screens.

FORWARD may also be entered as FOR or FO.

Optional Operands	
FCT	specifies the number of logical screens to move the session position by. If FCT is not specified, your session will be positioned forward one screen.

A logical screen is the number of lines of text shown in the current text display area -- it varies according to terminal type and whether SCREEN SPLIT is ON.

Use in a Procedure

FORWARD normally returns OK. (See SIBRETCD.)

Examples

Position forward 1 screen:

```
=> forward
```

Position forward 4 screens:

```
=> fo 4
```

FRENAME

Use FRENAME to change the name of a member. FRENAME is also used to move a member from one library to another.

FRENAME may also be entered as FREN.

Required Operands	
SRC	is the existing member name.
DEST	is the new member name.
PSWD	If the member is password protected, the password must be entered. The password may be up to 8 characters in length consisting of any characters. The renamed member remains password protected.
REPL	is used to specify whether the DEST member is to be replaced if a member by this name already exists. The default is "NO". If REPL=YES is specified and the DEST member does not exist the RENAME will be completed but the completion message will display the fact that the DEST member was not found. It is not possible to replace the following: <ul style="list-style-type: none">• a master member (i.e. check status is on)• a member that is part of a checkout relationship• a member that is password protected

FRENAME creates a display showing the required and optional operands that may be used to rename one member to a new member name.

If an asterisk (*) is entered as part of DEST, the asterisk is replaced with the corresponding part of SRC member name. This offers an easy way of moving a member from one library to another or adding a suffix or prefix to a name without keying the member name twice (see Examples).

FRENAME sets the "last referenced member" to DEST.

When Valid

FRENAME can only be used as an online command or in an online procedure. The user must have DEF access level for both the SRC library and the DEST library. SRC cannot be currently undergoing editing. SRC cannot be either the slave or the master in a checkout relationship.

Use in a Procedure

Return Codes:

OK	Successful.
CK	SRC member is part of a checkout relationship.
DP	DEST member already exists.
ED	SRC member is being edited.
LI	SRC member has active LIST session(s).

- NF One of the following:
- SRC member not found.
 - DEST library not found.

FRENAME sets the TXM variables to the attributes of DEST.

Examples

Request full screen RENAME prompt:

```
=> frename
```

Resulting display:

```
FRENAME
SOURCE MEMBER      :                               (required)
DESTINATION MEMBER :                               (required)
-----
PASSWORD           :
REPLACE OPTION     : NO
(PF1/PF13 = Update and End. PF2/PF14 Update and Redisplay. CLEAR/PF3 = End.)
```

FSESSION

Use FSESSION to alter session attributes.

FSESSION can also be entered as FSESS.

FSESSION has no operands and acts on the current session.

The FSESSION command sets attributes only for the current session. The current attributes of a session are displayed, on a MAPF screen, available to be overtyped. FSESSION is implemented as the procedure \$SYS.PROC.BIPFSSD which invokes the SCREEN and SESS commands to accomplish the requested changes.

Use in a Procedure

FSESSION Can only be used on-line or in an on-line procedure.

Session attributes can be accessed within a procedure using the SSD predefined variables.

Return codes

OK	Successful.
SV	Session is required.

Examples

Command entered:

```
=> fsess
```

Resulting display:

```
FSESS
SESSION NUMBER: 3, SESSION OBJECT: LIST LSL.APLDUMP

-----
CASE                :U
NULLS               :YES
FIND COLUMN         :1
SEQ NUMBER COL      :72-80
ZONE                :1-72
TAB COLUMNS        :1-10-16-32-72
ALT/DEF SCREEN      :D
LCA OFF/LEFT/RIGHT  :L
-----
```

```
(PF1, PF13 = Update and End. PF2, PF14 Update and Redisplay. CLEAR=End.)
```

GET

Use GET to insert a copy of the text of a member, \$LOG, \$MAIL, or \$STACK immediately following the current line in the current session.

Optional Operands	
MEM	MEM may be any member. MEM can also be specified as either \$LOG for the log, \$MAIL for the mail log, or \$STACK for the stack area. If MEM is not specified, the "last referenced member" is used.
BASE	specifies the starting line number for retrieval. If BASE is not specified, retrieval starts at the first line.
DEPTH	specifies the number of lines to retrieve. If DEPTH is not specified, retrieval continues to the end of the member, \$LOG, \$MAIL, or \$STACK.
PSWD	If the member is password protected, the password must be entered before get access is allowed. The password may be up to 8 characters in length consisting of any character.

GET \$STACK is the command line equivalent of the LCA I or B commands.

If the member is currently being edited, GET will retrieve the text of the member exclusive of any changes made in edit session.

GET sets the "last referenced member".

When Valid

For a GET from a library member, the user must have LIST access level to the MEM library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Member not found.
SC	Inadequate access level.

GET sets the TXM variables to the attributes of MEM.

Examples

Get the entire text of member OMRSYBT:

```
=> get omrsybt
```

Get the first 200 lines of member OMREXOP:

```
=> get omrexop,1,200
```

Get the text from the stack area:

```
=> get $stack
```

GETD (VSE version)

Use GETD to insert a copy of the text of a VSE sublibrary member immediately following the current line in the current session.

Optional Operands	
MEMD	the name and type of an existing VSE sublibrary member. The currently attached sublibrary (see ATTACHD) will be used unless the library and sublibrary are prefixed to the member name and type (i.e. library.sublib.member.type). If MEMD is not specified, the "last referenced VSE sublibrary member" will be retrieved.
BASE	specifies the starting line number for retrieval. If BASE is not specified, retrieval starts at the first line.
DEPTH	specifies the number of lines to retrieve. If DEPTH is not specified, retrieval continues to the end of the member.

GETD sets the "last referenced VSE sublibrary member".

When Valid

BIM-EDIT must be running in its own partition and under VSE/SP release 2.1 or later. The user must have LIST access level for the MEMD library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Member not found.
SC	Inadequate access level.

Examples

Insert the entire text of JRNLREC.C from the current sublibrary after the current line:

```
=> getd jrnlnrec.c
```

Insert lines 10 through 41 of DATECONV.A from sublibrary USERLIB3.ASM:

```
=> getd userlib3.asm.dateconv.a,10,41
```

GETD (MVS version)

Use GETD to insert a copy of the text of a Partitioned Data Set (PDS) member immediately following the current line in the current session.

Optional Operands	
MEMD	the name of the PDS member. The currently attached PDS (see ATTACHD) will be used unless the PDS name is prefixed to the member name, that is, pdsname.member or pdsname(member). If MEMD is not specified, the "last referenced PDS member" will be retrieved.
BASE	specifies the starting line number for retrieval. If BASE is not specified, retrieval starts at the first line.
DEPTH	specifies the number of lines to retrieve. If DEPTH is not specified, retrieval continues to the end of the member.

GETD sets the "last referenced PDS member".

GETD is allowed only for fixed block members with a record length 253 or less. Members containing source code generally meet this qualification, but load modules do not.

When Valid

The user must have LIST access level for the MEMD library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Member not found.
SC	Inadequate access level.

Examples

Insert the entire text of JRNLREC from the current PDS after the current line:

```
=> getd jrnlrec
```

Insert lines 10 through 41 of DATECONV from PDS BIM0001.V41:

```
=> getd bim0001.v41.dateconv,10,41
```

Insert the text of member PROG428 from PDS SYSLIB.PROBLEM:

```
=> getd syslib.problem(prog428)
```


GETP (VSE version)

Use GETP to insert a copy of the text of a POWER job entry immediately following the current line in the current session.

Optional Operands	
PRM1-4	<p>select the POWER job entry to be retrieved. The four operands allow you to specify the queue (RDR,LST,PUN,XMT), the job name, the job number, and the segment number. They can be specified in any order, except that segment, if specified, must follow job number. The following indicates which entry will be selected based on the operands specified:</p> <p>queue, job name, job number, segment</p> <p>The specified queue will be searched for the entry for the specified name, number, and segment.</p> <p>queue, job name</p> <p>The specified queue will be searched for the first entry for the specified name.</p> <p>queue, job number</p> <p>The specified queue will be searched for the first entry for the specified number, without regard to the job name.</p> <p>job number job name</p> <p>The LST queue will be searched for the first entry for the specified number or name.</p> <p>queue</p> <p>The specified queue will be searched for the first entry for the job name and number of the "last referenced POWER job entry"</p> <p>If PRM1-4 is not specified at all, the "last referenced POWER job entry" will be used.</p>
BASE	specifies the starting line number in the POWER job for the retrieval. If BASE is not specified, retrieval starts at the first line.
DEPTH	specifies the number of lines to retrieve. If DEPTH is not specified, retrieval continues to the end of the job.
USER	is site-defined. USER can be up to 16 characters in length. The exit routine BIXPWQA is provided addressability to this value.

Optional Operands (continued)	
CTL	<p>Must be specified as "YES" or "ON", "NO" or "OFF", or "SKIP". Specify "YES" or "ON" to retrieve the carriage control character as the first character of the line. Data characters are shifted one character to the right. If CTL is not specified or is specified as "NO", carriage control is not retrieved. All carriage control that results in carriage movement and no data will produce a blank line in the member. Specify "SKIP" to retrieve all printable lines skipping the carriage control lines that result in carriage movement only.</p> <p>Normally, System 370 printer channel commands are retrieved:</p> <ul style="list-style-type: none"> Hexadecimal 01 = Write without spacing Hexadecimal 09 = Space 1 line immediate Hexadecimal 0B = Write and Space 1 line Hexadecimal 11 = Write and Space 2 lines Hexadecimal 19 = Write and Space 3 lines Hexadecimal 73 = Block Data Check (causes no print action) Hexadecimal 8B = Skip to top of page immediate <p>See IBM documentation for additional values.</p> <p>Under certain circumstances, job entries are stored and will be retrieved with ASA carriage control. If this is the case, the following controls will be retrieved:</p> <ul style="list-style-type: none"> Plus (+) = Write without spacing Space () = Space 1 line before writing Zero (0) = Space 2 lines before writing Dash (-) = Space 3 lines before writing One (1) = Skip to top of page before writing <p>ASA carriage control is known to occur for job entries which are transferred from JES and for job entries generated by IBM's Document Management Facility and BIM-EDIT PRINT.</p>
TEXTF	<p>Specify "YES" or "ON" to indicate that text following mode is on for retrieval of input data from the POWER queue. When text following mode is on the four lines in sequence beginning with the four characters "++()" will be assembled into one long line before processing. The interpretation of the lines is as follows:</p> <ul style="list-style-type: none"> line 1: ++() text characters 1-64 line 2: ++() text characters 65-128 line 3: ++() text characters 129-192 line 4: ++() text characters 193-253 <p>This feature provides a capability to move long BIM-EDIT text lines through a narrow external medium and then reconstruct them. PUNCHF and REPROF produce lines in this format if their input is wider than 80 columns.</p>

GETP sets the "last referenced POWER job entry".

When Valid

Check with your System Administrator to determine the restrictions on access to POWER job entries at your site. The exit routine distributed with BIM-EDIT provides several restriction schemes your System Administrator can select among. (Use of and changes to the BIXPWQA exit routine are described in the System Reference Manual.)

Use in a Procedure

Return Codes:

OK	Successful.
NF	Job entry not found.
*	Codes as defined by BIXPWQA exit routine.

GETP sets the PWR variables to the attributes of the POWER job entry retrieved.

Examples

Insert a copy of "ASM100" from the list queue:

```
=> getp asml00
```

Insert a copy of "COMP", lines 200 - 649, from the punch queue:

```
=> getp pun,comp,base=200,depth=450
```

GETP (MVS version)

Use GETP to insert a copy of the text of JES data sets immediately following the current line in the current session.

Optional Operands	
JOB	selects the JES job to be inserted. If JOB contains letters, it specifies a jobname label from the Job Control Language JOB statement. If JOB is entirely digits, it specifies a job number assigned by JES. If not specified, the job from "last referenced JES data sets" will be used.
GROUP	<p>GROUP and DSET are mutually exclusive; if one is specified, the other must not be.</p> <p>selects the data sets to be inserted within the specified job. If GROUP is a single letter or digit, it specifies all data sets in the GROUP class in the JES "hold queue". If GROUP is "SYSOUT", it specifies all SYSOUT data sets. If GROUP is any other multiple character value, it specifies all data sets in the GROUP outgrp in the JES "output queue".</p> <p>An outgrp may be composed of three parts separated by periods in the form "name.qualifier.copy" where "name" is the JES or user-specified name, qualifier is a JES-assigned number, and copy is the copy number. All data sets matching a partial outgrp specification (for example, "name" only) will be inserted. To distinguish an outgrp with a one-character name from a class, follow the name with a period.</p>
DSET	<p>selects the data set to be inserted within the specified job. DSET can specify a user data set name from the ddname label of a Job Control Language DD statement. The data set name may be qualified by entering it in the form stepname.ddname. If the data set name is not unique within the job, the first instance will be used. You can specify JES pre-defined data sets, of which the most useful is \$JCL, which contains the Job Control Language for the job. DSET can also be specified as a JES-assigned data set number which is always unique. Use the INQUIREP command to display a list of data sets and their numbers.</p> <p>If none of JOB, GROUP and DSET are specified, the "last referenced data sets" are inserted. If only JOB is specified, all SYSOUT data sets are inserted.</p>
BASE	specifies the starting line number in the JES data sets to insert. If BASE is not specified, insertion starts with the first line
DEPTH	specifies the number of lines to insert. If DEPTH is not specified, insertion continues to the end of the data sets.

Optional Operands (continued)	
CTL	<p>Specify "YES" or "ON" to retrieve the carriage control character as the first character of the line. Data characters are shifted one character to the right. If CTL is not specified, carriage control is not retrieved. (ASA carriage control is retrieved:</p> <p>Plus (+) = Write without spacing Space () = Space 1 line before writing Zero (0) = Space 2 lines before writing Dash (-) = Space 3 lines before writing One (1) = Skip to top of page before writing</p> <p>See IBM documentation for additional values.)</p>
TEXTF	<p>Specify "YES" or "ON" to indicate that text following mode is on for retrieval of input data from the JES dataset. When text following mode is on the four lines in sequence beginning with the four characters "++()" will be assembled into one long line before processing. The interpretation of the lines is as follows:</p> <p>line 1: ++() text characters 1-64 line 2: ++() text characters 65-128 line 3: ++() text characters 129-192 line 4: ++() text characters 193-253</p> <p>This feature provides a capability to move long BIM-EDIT text lines through a narrow external medium and then reconstruct them. PUNCHF and REPROF produce lines in this format if their input is wider than 80 columns.</p>

GETP sets the "last referenced JES data sets".

When Valid

Check with your System Administrator to determine the restrictions on access to JES data sets at your site. The exit routine distributed with BIM-EDIT provides several restriction schemes your System Administrator can select among. (Use of and changes to the BIXPWQA exit routine are described in the System Reference Manual.)

Use in a Procedure

Return Codes:

- OK Successful.
- NF JES data sets not found.
- * Codes as defined by BIXPWQA exit routine.

GETP sets the JCT, JQE, and PDB variables to the attributes of the JES data sets retrieved. If GROUP is an outgrp, the JOE variables are also set.

Examples

Insert a copy of job "BIAS0010" SYSOUT data sets:

```
=> getp bias0010
```

Insert a copy of job 4789 \$JCL data set lines 200 - 649:

```
=> getp 4789,dset=$JCL,base=200,depth=450
```

Insert a copy of last referenced job output queue outgrp 3.1 data sets:

```
=> getp group=3.1
```

GETQ

Use GETQ to insert the text of all messages in the incoming-mail queue immediately following the current line in the current session. GETQ also deletes the messages.

GETQ has no operands.

GETQ provides an alternative to OPEN. OPEN displays a single message. GETQ retrieves all incoming messages and places them in the current session.

Messages are inserted in reverse order; the most recently received message will be at the top.

Use in a Procedure

GETQ normally returns OK. (See SIBRETCD.)

GETQ is implemented as the system procedure BIPGETQ.

Example

Get all incoming messages:

```
=> getq
```

GROUP

Use GROUP to select which of your session groups will be the current group of sessions.

GROUP may also be entered as GRP.

Optional Operands	
OPT	Values for OPT may be: + or N indicating the next group on the group chain. - or P indicating the previous group on the group chain. A - I specifying a group letter from A-I. If OPT is not specified, the next group of sessions of your allowed groups will be displayed.

Session groups are maintained on a chain. Any session group may be displayed at any time by use of the GROUP command.

The field "SESS=" on the information line shows three values. The first value, identifies which group of sessions is currently being displayed. The second value, identifies the session within the current group that is being displayed. The third value, within parentheses, is the total number of active sessions within the current group. For example, "SESS=A 1(4)" denotes that the current session is the first of four sessions within group A.

Use in a Procedure

GROUP normally returns OK. (See SIBRETC D.)

A syntax peculiarity occurs if "GROUP -" is used in a procedure. The space dash (" -") that terminates the command will be taken as a line continuation, usually leading to a command error. The simplest way to avoid the error is to use "GROUP '-'", or "GROUP P".

Examples

Switch to the next session group in chain:

```
=> group
```

Switch to the previous session group in chain:

```
=> grp -
```

Switch to the first session group:

```
=> group A
```


HELP

Use HELP to create a session display of information about BIM-EDIT commands, variables and topics.

To display a menu of commands or topics about the current session simply enter HELP. To display information about a particular topic or command, enter HELP followed by the topic name or command name. If the current display is a menu of commands or topics, enter HELP and move the cursor down to the desired subject line.

Optional Operands	
SUBJ	<p>is the subject for which you require assistance. SUBJ can be a BIM-EDIT command name or one of the following:</p> <p>MENU - Displays a menu of all topics or session types. ALL - Displays a list of all commands available for BIM-EDIT. cmnd - A BIM-EDIT command. L-x - A BIM-EDIT LCA command prefixed with "L-". VARS - A list of predefined variable groups. xxx - A group of predefined variables, for example, MMP. LCA - A list of LCA commands. UPD - A list of the help available for the User Developed Procedures contained in library \$SYS.UDP. nnx - The value "EDIT-nnx" or nnx, where "nnx" is the BIM-EDIT release, to display information pertinent to the enhancements, and problem fixes for that release. TOPIC - Displays a menu of the topic documentation available topic - Specify the name of a topic to view the documentation</p> <p>If SUBJ is not specified, a help menu for the current session type is listed.</p>

Use in a Procedure

HELP normally returns OK. (See SIBRETCO.)

HELP is implemented as the system procedure BIPHELP. HELP works by displaying text from a special BIM-EDIT library. Modifying the HELP information is described in Chapter 11, Customization, in the BIM-EDIT System Reference Manual.

Examples

Display information about the CHANGE command:

```
=> help change
```

List the available commands:

```
=> help
```

Display information about the LCA J command:

```
=> help l-j
```

Display information about the SIB predefined variables:

```
=> help sib
```

HOLD (VSE version)

Use HOLD to put a POWER job entry on hold, that is, to modify it in such a way as to suppress the automatic initiation of processing of the entry.

Optional Operands	
PRM1-4	<p>select the POWER job entry to be altered. The four operands allow you to specify the queue (RDR,LST,PUN,XMT), the job name, the job number, and the segment number. They can be specified in any order, except that segment, if specified, must follow job number. The following indicates which entry will be selected based on the operands specified:</p> <p>queue, job name, job number, segment</p> <p>The specified queue will be searched for the entry for the specified name, number, and segment.</p> <p>queue, job name</p> <p>The specified queue will be searched for the first entry for the specified name.</p> <p>queue, job number</p> <p>The specified queue will be searched for the first entry for the specified number, without regard to the job name.</p> <p>job number job name</p> <p>The LST queue will be searched for the first entry for the specified number or name.</p> <p>queue</p> <p>The specified queue will be searched for the first entry for the job name and number of the "last referenced POWER job entry"</p> <p>If PRM1-4 is not specified at all, the "last referenced POWER job entry" will be used.</p>

HOLD can also be invoked by entering the H command in the LCA area of a LIBRARYP display.

To take a job entry off hold, use RELEASE.

HOLD sets the "last referenced POWER job entry".

When Valid

Check with your System Administrator to determine the restrictions on access to POWER job entries at your site. The exit routine distributed with BIM-EDIT provides several restriction schemes your System Administrator can select among. (Use of and changes to the BIXPWQA exit routine are described in the System Reference Manual.)

Use in a Procedure

Return Codes:

OK	Successful.
NF	Job entry not found.
PW	Rejected by POWER.
*	Other codes as set by BIXPWQA exit routine.

HOLD sets the PWR variables to the attributes of the POWER job entry modified.

Examples

Hold an entry in the LST queue, identifying it with queue and job name:

```
=> hold lst,asm100
```

Hold an entry in the RDR queue, identifying it with queue and job number:

```
=> hold rdr,1278
```

Hold an entry in the LST queue, identifying it with queue, name, and number:

```
=> hold lst,xmllist,2700
```

Hold an entry in the LST queue, identifying it with just the job number:

```
=> hold 3426
```

Hold the last POWER job entry:

```
=> hold
```

HOLD (MVS version)

Use HOLD to put a JES job or data sets on hold, that is, to suppress automatic initiation of the job or data sets by JES.

Optional Operands	
JOB	selects the JES job to be altered. If JOB contains letters, it specifies a jobname label from the Job Control Language JOB statement. If JOB is entirely digits, it specifies a job number assigned by JES. If not specified, the job from "last referenced JES data sets" will be used.
GROUP	<p>selects the data sets to be altered within the specified job. To alter a JES job (as opposed to data sets), do not specify GROUP.</p> <p>GROUP specifies an outgrp in the JES "output queue". An outgrp may be composed of three parts separated by periods in the form "name.qualifier.copy" where "name" is the JES or user-specified name, qualifier is a JES-assigned number, and copy is the copy number. All data sets matching a partial outgrp specification (for example, "name" only) will be held. To distinguish an outgrp with a one-character name from a class, follow the name with a period.</p> <p>If neither JOB or GROUP is specified, the "last referenced data sets" are held. (If "last referenced data sets" was specified with a DSET operand, HOLD will be rejected because JES provides no facility to hold by data set.)</p>

To put a JES job (as opposed to data sets) on hold, specify the JOB operand but do not specify the GROUP operand. To usefully HOLD a JES job, the job must not have started processing.

HOLD operates on all data sets within the job that meet the selection.

The relationship between the "output queue" and the "hold queue" is a little confusing. For a data set to be in the "hold queue", HOLD=YES must have been specified or implied by the CLASS on the DD statement that created it and it must never have been RELEASEd or altered to a CLASS which implies HOLD=NO. When a HOLD command is issued for a data set that is in the "output queue", the data set is placed on hold status but remains in the "output queue", and must therefore be accessed by outgrp rather than by class.

HOLD can also be invoked by entering the H command in the LCA area of a LIBRARYP display.

To take a job or data sets off hold, use RELEASE.

HOLD sets the "last referenced JES data sets".

When Valid

Check with your System Administrator to determine the restrictions on access to JES data sets at your site. The exit routine distributed with BIM-EDIT provides several restriction schemes your System Administrator can select among. (Use of and changes to the BIXPWQA exit routine are described in the System Reference Manual.)

Use in a Procedure

Return Codes:

OK	Successful.
NF	JES job or data sets not found.
*	Other codes as set by BIXPWQA exit routine.

HOLD sets the JCT, JOE, and PDB variables to the attributes of the JES job or data sets modified. If GROUP is an outgrp, the JOE variables are also set.

Examples

Put the last referenced data sets on hold:

```
=> hold
```

Put outgrp 2 of job 1934 on hold:

```
=> hold 1934,2.
```

INCLUDE

Use INCLUDE to incorporate lines from a member during processing. INCLUDE is especially useful in a member being processed by a SUBMIT or COMPILE command.

INCLUDE may also be entered as INCL.

Required Operands	
MEM	is a library member. If specified without a library name, BIM-EDIT will retrieve MEM from the library for the member containing the INCLUDE, not the currently attached library. However, if the member containing the INCLUDE is the slave member in a checkout relationship, BIM-EDIT will retrieve MEM from the master library, not the slave library. To INCLUDE a member from the slave library, specify the library name as a part of MEM.

Optional Operands	
BASE	specifies the starting line number for retrieval. If BASE is not specified, retrieval starts at the first line.
DEPTH	specifies the number of lines to retrieve. If DEPTH is not specified, retrieval continues to the end of the member.

An included member can itself INCLUDE from another member. This nesting process is limited to 6 deep.

INCLUDEs are often used while submitting a job for processing. In this usage it is necessary to prefix the INCLUDE command with either a right parenthesis ")" or a slash "/" so that BIM-EDIT can distinguish it as a command rather than normal text. (The right parenthesis, otherwise known as the TRAP character, can only be used if the predefined variable PPDTRAP has a value of either "1" or "2" and the INCLUDE is being used within a "text-following" type command. The SUBMIT procedure supplied with BIM-EDIT sets PPDTRAP to "1", but it may be modified by your System Administrator.) Any data on the INCLUDE statement that is considered a comment must be preceded by a semi-colon (;).

If the member is currently being edited, INCLUDE will retrieve the text of the member ignoring any changes made in edit session.

(This description omits INCLUDE features which are used to incorporate data from sources other than members. INCLUDE is discussed completely in Chapter 3, Advanced User Commands, in the BIM-EDIT System Reference Manual.)

When Valid

INCLUDE can only be used in an online procedure, batch utility, or the application interface. If MEM is specified as a library member, the user must have LIST access level for the MEM library.

INCLUDE

Use in a Procedure

INCLUDE normally returns OK. (See SIBRETCD.)

Example

Often a number of programs need access to the same record descriptions. INCLUDE commands can be used to incorporate the members containing the record descriptions when they are submitted. These are the first few lines of a program structured this way:

```
=>
LIST 1982.OMCRSEQ                      SESS=A 1( 1) LINE=    0( 218)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----
*====* -- TOP OF MEMBER --
*====* OMCRSEQ: PROC;
*====*
*====* /INCLUDE OMBSIB
*====* /INCLUDE OMBSSED
*====*      SSEDPTR = SIBSESSP;
*====* /INCLUDE OMBRRH
*====*      RRHPTR = SIBRRHP;
```


IND\$FILE

The IND\$FILE command provides limited File Transfer support between BIM-EDIT and your PC using most PC 3270 Emulator packages.

The IND\$FILE command cannot be entered directly by the terminal user, it must be invoked using the 'Transfer' menu on your PC emulator screen.

(IBM Personal Communications uses the 'Transfer' menu, other emulators may use another menu.)

The IND\$FILE command of BIM-EDIT has the following requirements:

- Your PC must be connected to the host system running BIM-EDIT via a DFT type connection. This includes most LAN connections. CUT connections are not supported.
- Your PC emulator must support the DFT implementation of the IND\$FILE protocol. This uses 3270 structured fields for data transmission.
- You must be logged onto BIM-EDIT using a direct VTAM connection. CICS, BTAM, and TSO connections are not supported. Most TELNET connections are also considered by BIM-EDIT to be direct VTAM connections.

Send File To Host:

This transfer function will transfer all data from the specified PC dataset to the current BIM-EDIT session, following the current line. This is similar to the GET, GETD, GETP and GETI commands of BIM-EDIT.

The current session must be an updateable type session.

Most emulators will require that you enter a host file name, but it is ignored by BIM-EDIT.

Send file Transfer Options Supported:

BINARY	the data is transfered as is, no translation is performed.
ASCII	the data is translated from ASCII to EBCDIC.
CRLF	standard CRLF sequences are assumed to be in the data and will be used to determine where each line break occurs. If no CRLF is detected with 253 bytes, an automatic line break will occur. The following ASCII sequences will cause a line break: X'0D0A', X'0D', X'0C'
NOCRLF	each line is transfered to the PC using the length specified for the LRECL option.
LRECL=n	used with the NOCRLF option to determine the length of each line in the PC file. The maximum that can be specified is 253. The minimum that can be specified is 1.

The LRECL option can be entered as: LRECL=nnn
 LRECL(nnn)
 LRECL nnn

All other options are ignored. If an option is specified more than once, or conflicting options are specified, such as specifying both BINARY and ASCII, the last option takes affect.

Receive File From Host:

This transfer function will transfer all lines from the current BIM-EDIT session, beginning with the current line, to the specified PC dataset.

The current session can be any BIM-EDIT session.

Most emulators will require that you enter a host file name, but it is ignored by BIM-EDIT.

Receive File Transfer Options Supported:

BINARY	the data is transfered as is, no translation is performed.
ASCII	the data is translated from EBCDIC to ASCII.
CRLF	the standard CRLF sequence is inserted after each lineof the BIM-EDIT session.
NOCRLF	each line is transfered to the PC using the length specified for the LRECL option. Longer lines will be truncated, shorter lines will be padded with blanks.
LRECL=n	used with the NOCRLF option to determine the length of each line in the PC file. The maximum that can be specified is 253. If zero is specified, the data is sent to the PC with no trailing blanks.

The LRECL option can be entered as: LRECL=nnn
 LRECL(nnn)
 LRECL nnn

All other options are ignored. If an option is specified more than once, or conflicting options are specified, such as specifying both BINARY and ASCII, the last option takes affect.

INQUIRE

Use INQUIRE to display the attributes of a member.

INQUIRE may also be entered as INQ.

Optional Operands	
MEM	is the member name to be queried. If MEM is not specified, the "last referenced member" is used.

(FALTER can also be used to display a member's attributes and allows changing them by overtyping on the display.)

INQUIRE sets the "last referenced member".

When Valid

INQUIRE can only be used in an online command or in an online procedure. The user must have LIST access level for the MEM library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Member not found.
SC	Inadequate access level.

INQUIRE sets the TXM variables to the attributes of MEM.

INQUIRE is implemented as the system procedure BIPINQM.

Example

Inquire about member ESG1030:

```
=> inquire esgl030
```

INQUIRED (VSE only)

Use INQUIRED to display the attributes of a VSE library member.

INQUIRED may also be entered as INQD.

Optional Operands	
MEMD	is the member name to be queried. If MEMD is not specified, the "last referenced VSE mbr" is used.

When Valid

INQUIRED can only be used in an online command or in an online procedure. The user must have LIST access level for the MEMD library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Member not found.
SC	Inadequate access level.

INQUIRED is implemented as the system procedure BIPINQD.

Example

Inquire about member IJSYSRS.SYSLIB.SPLEVEL.PROC

```
=> inquired ijsysrs.syslib.splevel.proc
```

INQUIREP (VSE)

Use INQUIREP to display a list of POWER LST and/or PUN queue entries for one or more currently executing jobs. When INQUIREP is invoked online, it creates a DISP session for the results.

INQUIREP may also be entered as INQP.

Optional Operands	
PART	specifies the partition for which entries are to be displayed. PART can be entered as either a 1 or 2 character value. If 1 character is entered, all partitions with the same starting character will be displayed. If omitted, all partitions will be displayed.
CUU	specifies the 3-character cuu of the entries to be displayed. If specified as 'LST', only LST entries will be displayed. If specified as 'PUN', only PUN entries will be displayed. If omitted, all LST and PUN entries will be displayed.
NAME	is the job name. If name is specified as "ALL", or is not specified, all jobs you have access to will be shown. All other names will be assumed to be generic. The name can start with an asterisk (*) to be compatible with POWER's syntax, but it is not required.
NUMBER	is the job number. If NUMBER is specified, only entries whose job number matches NUMBER will be shown.

INQUIREP can also be invoked by entering the "X" command in the LCA area of a currently executing RDR queue entry on a LIBRARYP display.

The LCA L command can be used on the INQUIREP display to start a LISTP session.

DISP sessions such as that produced by the INQUIREP command can be updated or "refreshed" by the REFRESH command. (You'll probably find it convenient to assign the REFRESH command to a PF key.)

The INQUIREP display has control information to the right of column 135 for use by LCA commands. If you use INQUIREP display lines in another context, you may need to remove the control information using the BLANK or KEEP commands.

When Valid

Check with your System Administrator to determine the restrictions on access to POWER job entries at your site. The exit routine distributed with BIM-EDIT provides several restriction schemes your System Administrator can select among. (Use of and changes to the BIXPWQA exit routine are described in the System Reference Manual.)

Use in a Procedure

Return Codes:

- OK Successful.
- * Other codes as set by BIXPWQA exit routine.

INQUIREP (VSE)

If INQUIREP is invoked in batch utility, the display destination is controlled by predefined variable SIBOUTPT.

Examples

Display a list of all entries for executing jobs in partition class C:

```
=> inquirep C
```

Display a list of all LST entries for executing jobs:

```
=> inquirep ,LST
```

Display a list of all entries for executing jobs:

```
=> inqpp
```

Output from INQUIREP

```
=>
DISP  -> inqp                                SESS=A 1( 1) LINE=    0    19)
-----1-----2-----3-----4-----5-----6-----7-----8-----9-----
***** -- TOP OF DISPLAY --
***** POWER ACTIVE QUEUE DISPLAY
*****
***** --POWER JOB -- C   CARDS/          DESTINATION
***** PP QUE, CUU NAME  NUMBER L  PAGES   FORM   USER INFO  NOE   USER   DATE   TIME
*****
***** F3 LST,02E VTAM      6834 A      0      BM01      BIM      R000    09/18/2000 00.05.13
***** F8 LST,02E BIMWNDOW  6838 A    6,194  BM01-BW45H  BIM      FKE     09/18/2000 00.06.24
***** Y1 LST,02E BIMTCPIP  6840 Q    5,850      BIM      SJA     09/18/2000 00.06.41
***** G1 LST,02E BIMTMAN   6839 Q      210  BIMTMAN.PROD.JCL BIM      R000    09/18/2000 00.07.09
***** G1 PUN,02D BIMTMAN   6839 Q      0      BIMTMAN.PROD.JCL BIM      R000    09/18/2000 00.07.09
***** V2 LST,02E BIMTCPT   6841 Q     136      BIM      SJA     09/18/2000 00.07.33
***** G2 LST,02E JCLSCHEDE 6843 A      0      BIM      SJA     09/18/2000 00.07.46
***** F4 LST,02E CICSPROD  6842 A    2,039  BM01      BIM      R000    09/26/2000 21.12.25
***** F4 PUN,02D CICSPROD  6842 A      0      BM01      BIM      R000    09/26/2000 21.12.25
***** T1 LST,02E CICSTEST  8738 A    1,011      BIM      R000    09/28/2000 21.47.45
***** F9 LST,02E BIMEDIT   9214 A     131  EDPD + ED54AD BIM      R000    10/03/2000 19.20.26
```

INQUIREP (MVS)

Use INQUIREP to display a list of JES data sets within a JES job. When INQUIREP is invoked online, it creates a DISP session for the results.

INQUIREP may also be entered as INQP.

Optional Operands	
QUEUE	specifies the JES queue. QUEUE may be specified as "I" for the input queue, "H" for the held queue or "O" for the output queue.
JOB	selects the JES job to be displayed. If JOB contains letters, it specifies a jobname label from the Job Control Language JOB statement. If JOB is entirely digits, it specifies a job number assigned by JES. If not specified, the job from "last referenced JES data sets" will be used.

INQUIREP can also be invoked by entering the "X" command in the LCA area of a LIBRARYP display.

All data sets for the JES job are listed individually and their data set numbers are given. The data set number can be useful for the GETP, LISTP and READP commands.

Use LIBRARYP to list multiple JES jobs and summarize their data sets. Use FALTERP to display and modify attributes of JES data sets.

The LCA L command can be used on the INQUIREP display to start a LISTP session.

INQUIREP sets "last referenced JES data sets".

The INQUIREP display has control information to the right of column 135 for use by LCA commands. If you use INQUIREP display lines in another context, you may need to remove the control information using the BLANK or KEEP commands.

When Valid

Check with your System Administrator to determine the restrictions on access to JES data sets at your site. The exit routine distributed with BIM-EDIT provides several restriction schemes your System Administrator can select among. (Use of and changes to the BIXPWQA exit routine are described in the System Reference Manual.)

Use in a Procedure

Return Codes:

OK	Successful.
NF	JES job not found.
*	Other codes as set by BIXPWQA exit routine.

INQUIREP sets the JQE, JCT and PDB variables to the attributes of the JES job displayed.

If INQUIREP is invoked in batch utility, the display destination is controlled by predefined variable SIBOUTPT.

Examples

Display a list of all data sets in job U729842:

```
=> inquirep U729842
```

Display a list of all data sets in the last referenced job:

```
=> inqp
```

Output from INQUIREP

```
=>
DISP  -> inqp                                SESS=A 1( 1) LINE=    0(   15)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----
*====* -- TOP OF DISPLAY --
*====* BICEDTD/8083 DATASETS
*====*
*====* DSET      STEP      PROC      DDNAME  LINES  C
*====* NMBR              STEP              L
*====* -----
*====* 1          $JCL          31
*====* 2 JES2          $JES2LOG  17
*====* 3 JES2          $JCLIMG   31
*====* 4 JES2          $SYMSGS   50 A
*====* 5 JES2          $INTTEXT   0
*====* 6          $JOURNAL   0
*====* 101 ASM        SYSIN     1990
*====* 102 LKED              1
*====* 103 ASM        SYSPRINT  229 A
*====* 104 LKED        SYSPRINT  20 A
*====* -- END OF DISPLAY --
```


INSERTI

Use INSERTI to insert a specified line after the current line of the current session.

INSERTI may also be entered as INSI, INSERT, or INS.

Required Operands	
LINE	is the line to insert.

The inserted line becomes the current line.

Use in a Procedure

INSERTI normally returns OK. (See SIBRETCDD.)

Example

EDIT session before INSERTI command

```
=> inserti ' /* customer record */'
EDIT 2169.BX2139                      SESS=A 1( 1) LINE=      1(      6)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*      DCL
*====*      CF INV N                      PIC 'ZZZZZZ',
*====*      CF ALLOC BAL                  PIC 'ZZZZZZ9V.99',
*====*      CF UNALLOC BAL                PIC 'ZZZZZZ9V.99',
*====*      CF ORD BAL                   PIC 'ZZZZZZ9V.99',
*====*      CF CR BAL                     PIC 'ZZZZZZ9V.99',
*====*      -- END OF MEMBER --
```

EDIT session after INSERTI command

```
=>
EDIT 2169.BX2139                      SESS=A 1( 1) LINE=      2(      7)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*      /* CUSTOMER RECORD */
*====*      CF INV N                      PIC 'ZZZZZZ',
*====*      CF ALLOC BAL                  PIC 'ZZZZZZ9V.99',
*====*      CF UNALLOC BAL                PIC 'ZZZZZZ9V.99',
*====*      CF ORD BAL                   PIC 'ZZZZZZ9V.99',
*====*      CF CR BAL                     PIC 'ZZZZZZ9V.99',
*====*      -- END OF MEMBER --
```

JOIN

Use JOIN to combine two lines of text into one line of text.

The cursor can be used to designate the location where the JOIN is to occur or optionally the following parameters can be used. If the cursor is used the text starting at column 1 of the line following the line where the cursor is positioned will be appended to the right of the line starting at the cursor location. All data on the line to the right of the cursor will be overlayed by the appended data and lost. The cursor will remain at the location where the join occurred.

Required Operands	
LOC	<p>is the column where data will be appended, and may be a number or a character string:</p> <ul style="list-style-type: none">• If LOC is a number from 1 to 252, the text from the line after the current line will be appended to the current line at the column specified.• If LOC is a character string, the text from the line after the current line will be appended to the current line after the location where the string was found. The character string can be up to 36 characters long.
CASE	<p>specifies whether upper/lower case should be considered when determining matches. If specified as "U", case is ignored ("the" matches "The"). If specified as "M", both case and letters must match. If CASE is not specified, "U" is assumed.</p>

Use in a Procedure

JOIN always returns OK.

Example

Note: In the following examples the underscore "_" designates the cursor.

EDIT session before JOIN command

```
=> join
EDIT 1283.CPACT                      SESS=A 1( 1) LINE= 17( 20)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==*      SEE THE QUICK BROWN FOX
*==*      JUMP OVER THE LAZY BROWN DOG
*==*      SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==*      SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
-- END OF MEMBER --
```

EDIT session after JOIN command

```
=> join
EDIT 1283.CPACT                      SESS=A 1( 1) LINE=   17(   19)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==*      SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==*      SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==*      SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
-- END OF MEMBER --
```

JUSTIFYL

Use JUSTIFYL to left justify current session text within a specified column range for a specified number of lines starting with the current line.

JUSTIFYL may also be entered as JUSTL.

Optional Operands	
FCT	specifies the number of lines for which text is to be left justified. If FCT is specified as an asterisk (*), text will be left justified for all of lines through the end of the session. If FCT is not specified, text will be left justified for the current line only.
ZONE	is the column range in which to left justify text. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to left justify text within columns 11 to 66, specify ZONE as "11-66". ZONE=5 is the same as ZONE=5-5, ZONE=5-* is the same as ZONE=5-253 and ZONE=-5 is the same as ZONE=1-5. If ZONE is not specified, text is left justified in the current session zone (see the ZONE operand of the DEFINE, ALTER, and SESS commands.)

JUSTIFYL will left justify text; it will align text on the left-hand side of the ZONE. In effect, all leading blanks are deleted.

JUSTIFYL is the command line equivalent of the LCA (command.

Use in a Procedure

JUSTIFYL normally returns OK. (See SIBRETCDD.)

Example

Left justify text within column range 18 to 58 for 3 lines:

EDIT session before JUSTIFYL command

```
=> justifyl 3,18-58
EDIT  IS.361-100                      SESS=A 1( 1) LINE=    56( 3125)
-----1-----2-----3-----4-----5-----6-----7--
*==== HD(1).VAR = ' JENSON & HALLORAN, INC. ' ;
*==== HD(2).VAR = ' CONSOLIDATED STATEMENT OF OPERATIONS ' ;
*==== HD(3).VAR = ' AS OF DECEMBER 31, 1995 ' ;
```

EDIT session after JUSTIFYL command

```
=>
EDIT  IS.361-100                      SESS=A 1( 1) LINE=    56( 3125)
-----1-----2-----3-----4-----5-----6-----7--
*==== HD(1).VAR = 'JENSON & HALLORAN, INC. ' ;
*==== HD(2).VAR = 'CONSOLIDATED STATEMENT OF OPERATIONS ' ;
*==== HD(3).VAR = 'AS OF DECEMBER 31, 1995 ' ;
```

JUSTIFYR

Use JUSTIFYR to right justify current session text within a specified column range for a specified number of lines starting with the current line.

JUSTIFYR may also be entered as JUSTR.

Optional Operands	
FCT	specifies the number of lines for which text is to be right justified. If FCT is specified as an asterisk (*), text will be right justified for all of lines through the end of the session. If FCT is not specified, text will be right justified for the current line only.
ZONE	is the column range in which to right justify text. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to right justify text within columns 11 to 66, specify ZONE as "11-66". ZONE=5 is the same as ZONE=5-5, ZONE=5-* is the same as ZONE=5-253 and ZONE=-5 is the same as ZONE=1-5. If ZONE is not specified, text is right justified in the current session zone (see the ZONE operand of the DEFINE, ALTER, and SESS commands.)

JUSTIFYR will right justify text; it will align text on the right-hand side of the ZONE.

JUSTIFYR is the command line equivalent of the LCA) command.

Use in a Procedure

JUSTIFYR normally returns OK. (See SIBRETC D.)

Examples

Right justify text within column range 18 to 58 for 3 lines:

EDIT session before JUSTIFYR command

```
=> justifyl 3,18-58
EDIT  IS.361-100                      SESS=A 1( 1) LINE=    56( 3125)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*****      HD (1) .VAR = '                JENSON & HALLORAN, INC.          ' ;
*****      HD (2) .VAR = '      CONSOLIDATED STATEMENT OF OPERATIONS      ' ;
*****      HD (4) .VAR = '                AS OF DECEMBER 31, 1995          ' ;
```

EDIT session after JUSTIFYR command

```
=>
EDIT  IS.361-100                      SESS=A 1( 1) LINE=    56( 3125)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*****      HD (1) .VAR = '                JENSON & HALLORAN, INC.          ' ;
*****      HD (2) .VAR = '      CONSOLIDATED STATEMENT OF OPERATIONS      ' ;
*****      HD (3) .VAR = '                AS OF DECEMBER 31, 1995          ' ;
```

KEEP

Use KEEP to replace current session text with blanks in all but a specified column range for a specified number of lines starting with the current line.

Optional Operands	
ZONE	is the column range to leave intact. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to blank out all but the columns 11 to 20, specify ZONE as "11-20". ZONE=5 is the same as ZONE=5-5, ZONE=5-* is the same as ZONE=5-253 and ZONE=-5 is the same as ZONE=1-5. If ZONE is not specified, all but the current session zone is blanked (see the ZONE operand of the DEFINE, ALTER, and SESS commands.)
FCT	specifies the number of lines to update. If FCT is specified as an asterisk (*), all of the lines through the end of the session will be updated. If FCT is not specified, only the current line will be updated.

BLANK can be used to blank out columns within a specified range.

Use in a Procedure

KEEP normally returns OK. (See SIBRETC D.)

Example

EDIT session before KEEP command

```
=> keep 6-13,*
EDIT  CFF.TEMP                      SESS=A 1( 1) LINE=    0(   72)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----
*====* -- TOP OF MEMBER --
*====* LST BIMWDX  1427 3 H A      5      1 1      BM01
*====* LST BIMWND  1446 3 H A     551     17 1
*====* LST BIMWNC  1620 3 H A   4,472    109 1
*====* LST BIMWNX  1448 3 H A     20      1 1      BM01
```

EDIT session after KEEP command

```
=>
EDIT  CFF.TEMP                      SESS=A 1( 1) LINE=    0(   72)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----
*====* -- TOP OF MEMBER --
*====*      BIMWDX
*====*      BIMWND
*====*      BIMWNC
*====*      BIMWNX
```

KEYS

Use KEYS to display and alter your function (PF) key settings.

KEYS may also be entered as KEY.

KEYS has no operands.

It is possible to have command line input data (up to four separate parameters) merged with PF key data before the command string is presented to BIM-EDIT. The parameter data insertion will take place when a PF key is depressed and the associated PF key data has a reverse quote (') imbedded within it. Any number of reverse quote marks can appear within the PF key definition. As each reverse quote mark is encountered in the PF key data, it is replaced with a parameter from the command line input. If only one parameter is entered on the command line, all reverse quote marks encountered in the PF key data will be replaced with that single parameter. If two command line parameters are entered and there are five occurrences of the reverse quote mark in the PF key data, parameter one from the command line will replace the first reverse quote mark and command line parameter two will replace reverse quote two through five. The processing option for the data insertion must be set to "ONLY".

PF keys 1-24 can have a processing option assign to them to control how they are processed by BIM-EDIT. The following options can be specified:

AFTER	PF key data will be processed after the command line.
BEFORE	PF key data will be processed before the command line.
IGNORE	PF key data will not be processed if there is data on the command line.
ONLY	Only PF key data will be processed. Data on the command line is used if the 'data insert' character is found. If the 'data insert' character is not found, the data on the command line is ignored.

If no processing option is specified, 'AFTER' is assumed.

When Valid

KEYS can only be used as an online command or in an online procedure.

Use in a Procedure

KEYS normally returns OK. (See SIBRETCOD.)

KEYS is implemented as the system procedure BIPKEYS.

Example

Modify function keys

=> keys

Screen display 1 of 2 after KEYS

KEYS QUERY / ALTER PF KEYS			(PF1 - PF12)
KEY	OPTION	DATA STRING	
PF1	AFTER	SCR AUD,TOG	
PF2	ONLY	REF	
PF3	AFTER	END	
PF4	ONLY	SAVE;PROC	
PF5	AFTER	SCR SP,TOG	
PF6	ONLY	OPEN	
PF7	AFTER	BACK	
PF8	AFTER	FOR	
PF9	AFTER	VIEW 1	
PF10	AFTER	VIEW 53	
PF11	ONLY	ROT +	
PF12	ONLY	ROT -	
(PF1, PF13 = UPDATE. CLEAR = END ALTER. PF8 = FORWARD.)			

Screen display 2 of 2 after KEYS

KEYS QUERY / ALTER PF KEYS			(PF13 - PA3)
KEY	OPTION	DATA STRING	
PF13	AFTER	BACK;NEXT 2	
PF14	ONLY	UP 10	
PF15	ONLY	TOP	
PF16	AFTER	FOR;UP 2	
PF17	ONLY	NEXT 10	
PF18	ONLY	BOT	
PF19			
PF20			
PF21			
PF22	AFTER	SCR ALT,TOG	
PF23			
PF24			
ENTER		NEXT	
CLEAR			
PA1			
PA2			
PA3			
(PF1, PF13 = UPDATE. CLEAR = END ALTER. PF7 = BACKWARD.)			

The settings displayed on these screens may be changed by overtyping them. Hit ENTER to set the new values.

LADD

Use LADD to add blank line(s) to the current session immediately following the current line.

LADD may also be entered as LA.

Optional Operands	
FCT	specifies the number of lines to add. If FCT is not specified, one line will be added.

The cursor will be positioned at the first new blank line. The current line does not change.

LADD is the command line equivalent of the LCA A command.

Use in a Procedure

LADD normally returns OK. (See SIBRETC D.)

Example

EDIT session before LADD command

```
=> ladd 2
EDIT 2169.BX2139                      SESS=A 1( 1) LINE=      1(      6)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*      DCL
*====*      CF INV BAL                  PIC 'ZZZZZZ9V.99',
*====*      CF ALLOC BAL                PIC 'ZZZZZZ9V.99',
*====*      CF UNALLOC BAL              PIC 'ZZZZZZ9V.99',
*====*      CF ORD BAL                  PIC 'ZZZZZZ9V.99',
*====*      CF CR BAL                   PIC 'ZZZZZZ9V.99',
*====*      -- END OF MEMBER --
```

EDIT session after LADD command

```
=>
EDIT 2169.BX2139                      SESS=A 1( 1) LINE=      1(      8)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*      DCL
*====*      _
*====*      CF INV BAL                  PIC 'ZZZZZZ9V.99',
*====*      CF ALLOC BAL                PIC 'ZZZZZZ9V.99',
*====*      CF UNALLOC BAL              PIC 'ZZZZZZ9V.99',
*====*      CF ORD BAL                  PIC 'ZZZZZZ9V.99',
*====*      CF CR BAL                   PIC 'ZZZZZZ9V.99',
*====*      -- END OF MEMBER --
```

LIBRARY

Use LIBRARY to display a list of members in a library. All members in the library can be listed, or only those members starting with specified characters. When LIBRARY is invoked online, it creates a DISP session for the results.

LIBRARY may also be entered as LIB.

Optional Operands	
MEM	is the library and the initial characters of the members to be listed. If no library is specified (i.e., MEM does not contain a period) members in the currently attached library are listed. If the library is specified but the member is not (i.e., MEM ends in a period), all members in the specified library are listed. If MEM is not specified, all members in the current library are listed. MEM can contain * and ? characters, which will match as described below in the Extended Search Pattern rules. If the library name part of MEM has * or ? characters, you can list members in more than one library.
FMT	<p>is the format of the display. The following are provided:</p> <p>NAME only the NAME is shown. This format is handy when you need to create a list of members for use in a procedure.</p> <p>NAME1 same as NAME.</p> <p>NAME2-NAME14 Only names are shown, the count per screen line determined by the number after 'NAME'. Four (NAME4) names will fit on an 80 column wide screen view, seven (NAME7) will fit on a 132 column view.</p> <p>DET the fields NAME, TYPE, ATTR, # text lines, # audit lines, date created, date updated, audit status, stamp status, update user, and title are shown on a single line. This format also displays a line of totals for both text and audit lines.</p> <p>FULL all of the fields specified in the DET format above are shown. In addition, the create user is shown. FULL is most useful with a 132 character screen or when writing the output to a printer. Totals are provided as with the DET format.</p>

Optional Operands (continued)	
	<p>CHECK a display designed for checkout / checkin management is created. The fields NAME, checkout status, checkout library, checkout date, audit status, stamp status, date created, and date updated are shown.</p> <p>STATS a display showing only NAME, BYTES, LINES, CI's, TITLE, and last update date and user.</p> <p>If FMT is not specified, the default format shows NAME, TYPE, and TITLE on a single line.</p>
TYPE	If TYPE is specified, only members which match the specified TYPE are displayed. If the value of TYPE begins with a backslash character (\), matching will use Extended Search Pattern rules, as described below. See the DEFINE command for information about TYPE.
ATTR	If ATTR is specified, only members which match the specified ATTR are displayed. If the value of ATTR begins with a backslash character (\), matching will use Extended Search Pattern rules, as described below. See the DEFINE command for information about ATTR.
CKUSER	If CKUSER is specified, only members that have been checked out to the CKUSER user are displayed. If CKUSER is specified as an asterisk (*), all members that have been checked out are displayed. If the value of CKUSER begins with a backslash character (\), matching will use Extended Search Pattern rules, as described below.
TITLE	If TITLE is specified, only members that contain its value in their TITLES are displayed. If the value of TITLE begins with a backslash character (\), matching will use Extended Search Pattern rules, as described below.

Normally members from the currently attached library are listed. However, members from another library can be listed by providing the library name as part of MEM. For example, to display all members from library TM20 that start with the three characters "BIC", enter:

```
=> library tm20.bic
```

To display all TM20 members, enter:

```
=> library tm20.
```

The following LCA commands used in conjunction with the LIBRARY display can be quite useful:

E	EDIT	Q	FALTER
L	LIST	S	PROCESS
P	PURGE		

See Chapter 5, LCA Commands, in the BIM-EDIT User Reference Manual for more information.

All LIBRARY display formats except NAME have control information to the right of column 135 for use by LCA commands. If you use LIBRARY display lines in another context (i.e., as input to a procedure) you may need to remove the control information using the BLANK or KEEP commands.

If your terminal is running under control of SNA, you may interrupt a long LIBRARY command by pressing the ATTN key.

Several operands of the LIBRARY command can be treated as Extended Search Patterns, in which certain characters have special meanings:

- ? matches any single character
- * matches any sequence of zero or more characters
- @ matches zero or more instances of the character that follows the @
- ! matches the beginning of the operand
- < matches the beginning of the operand or any non-alphanumeric character
- > matches the end of the operand or any non-alphanumeric character
- | separates two patterns, EITHER of which may match for a successful search
- + separates two patterns, BOTH of which must match for a successful search
- ~ (NOT) reverses line-by-line success/failure of the pattern following it
- \ suppresses the special meaning of the character that follows the \

See Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual for more information about Extended Search Patterns.

When Valid

The user must have EXEC access level for the MEM library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Library not found.
SC	Inadequate access level.

If LIBRARY is invoked in batch utility, the display destination is controlled by predefined variable SIBOUTPT.

Examples

Default Format

```
=>
DISP  -> lib bir                                SESS=A 1( 1) LINE=    0( 439)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF DISPLAY --
*====* LIBRARY=ED51A
*====* -----
*====* MEMBER          TYPE          TITLE
*====* -----
*====* BIRAPEX          ASM          ROOT MODULE
*====* BIRAPFL          ASM          FLUSH APPL INTF OUTPUT BUFFER
*====* BIRAPPL          ASM          APPLICATION DIALOGUE CONTROL
*====* BIRAPRD          ASM          RECEIVE LINE FROM APPLICATION
```

NAME Format

```
=>
DISP  -> lib bir*-d,name                        SESS=A 1( 1) LINE=    0( 127)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF DISPLAY --
*====* BIRDLAY$-D
*====* BIRDPEJ$-D
*====* BIRDPLU$-D
*====* BIRDPML$-D
*====* BIRDPSS$-D
*====* BIRDPVT$-D
```

DET Format

```
=>
DISP  -> lib ed51?.bicbkpg,det,type=\asm*        SESS=A 1( 1) LINE=    0( 19)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF DISPLAY --
*====* LIBRARY=ED51A
*====* -----
*====* MEMBER          TYPE          ATTR    --- LINES ---          DATE -----
*====*                TEXT  AUDIT    CREATE  UPDATE
*====* BICBKPG          ASM          NLRLH      36    233  08/09/1989 04/24/1995
*====*                36    233
*====* LIBRARY=ED51L
*====* -----
*====* MEMBER          TYPE          ATTR    --- LINES ---          DATE -----
*====*                TEXT  AUDIT    CREATE  UPDATE
*====* BICBKPG          ASMLIST    04/24/95    88      0  02/13/1990 04/24/1995
*====*                88      0
*====*                124    233
*====* -- END OF DISPLAY --
```

CHECK Format

```
=>
DISP  -> lib ed51a.,check,ckuser=mwd            SESS=A 1( 1) LINE=    0( 7)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF DISPLAY --
*====* LIBRARY=ED51A
*====* -----
*====* MEMBER          ST      USER          CHECKOUT          LAST          AUD
*====*                ST      USER          LIBRARY          DATE          UPDATE  ST
*====* BIZSYRF          MS      MWD          BIM              12/21/1995 08/09/1989 ON
*====* BIZUSRF          MS      MWD          BIM              12/21/1995 08/09/1989 ON
```

Legend: CHECKOUT ST indicates CHECKOUT status. It can be MS=master, SL=slave, ON=CHECK is ON, OF=CHECK is OFF. AU ST indicates AUDIT status. ST ST indicates STAMP status.

STATS Format

```
=>
DISP -> lib bir,stats                                SESS=A 1( 1) LINE=    0( 903)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF DISPLAY --
*====* LIBRARY=ED00A
*====* MEMBER          TYPE      TEXT    TEXT    AUDIT   TEXT  AUDIT  AUD  UPDATE
*====*                BYTES    LINES   LINES   CI     CI    ST   USER
*====* -----
*====* BIRABFR$-M      ASM       4564    125    320     2     3 ON  LSL
*====* BIRADVF$-D      ASM       2026     62     0      1     0 ON  PG
*====* BIRADVF$-M      ASM       2736     80     0      1     0 ON  PG
*====* BIRAPEX         ASM       6334    195    310     2     3 ON  PG
*====* BIRAPFL         ASM       1047     39     0      1     0 ON  GJB
*====* BIRAPPL         ASM       4234    119     0      2     0 ON  GJB
*====* BIRAPRD         ASM       2289     76     0      1     0 ON  GJB
*====* BIRAPWR         ASM       1426     51     0      1     0 ON  PG
```

LIBDS (VSE only)

Use LIBDS to create a session displaying the output of a VSE LIBR SEARCH command.

The operands for the LIBDS command are those supported by the LIBR SEARCH command. BIM-EDIT does not edit them. All operands specified following the LIBDS command are passed directly to LIBR.

The syntax of the SEARCH command is summarized here, but can also be found in the VSE/ESA System Control Statements manual.

The following command formats are supported:

- `Libds mn.mt Lib=lib1 lib2 ...`
- `Libds mn.mt Lib=*`
- `Libds mn.mt Sublib=lib.sub1 lib.sub2 ...`
- `LIBDS mn.mt LIBDef=xxxxxxx LIBUse=xxxxxxx Partition=xx`

where:

`mn.mt`

Specifies the member name and member type of the member to be searched. The specification may be generic

`Lib=`

Specifies the library or libraries in which the member is to be searched.

`Lib=*`

Specifies that the member will be searched in all libraries which are currently open in the system.

`Sublib=`

Specifies the sublibrary or sublibraries in which the member is to be searched.

`LIBDef=PHASE | SOURCE | OBJECT | PROC | DUMP`

Indicates that the member is to be searched in the active LIBDEF chain of the specified type. (The LIBDEF chain is defined in the job control LIBDEF command). The LIBDEF chain can be further identified with the LIBUSE and the PARTITION operands.

`LIBUse=SEARCH | CATALOG`

Specifies whether the SEARCH or the CATALOG library list of the job control LIBDEF command is to be searched.

`Partition=partid`

Indicates the partition in which the specified LIBDEF chain lies. The default is the BIM-EDIT partition.

The following LCA commands used in conjunction with the LIBDS display can be quite useful:

```
E  EDITD
L  LISTD
P  PURGED
```

See Chapter 5, LCA Commands, in the BIM-EDIT User Reference Manual for more information.

The LIBDS display has control information to the right of column 135 for use by LCA commands. If you use LIBDS display lines in another context, you may need to remove the control information using the BLANK or KEEP commands.

When Valid

BIM-EDIT must be running in its own partition and under VSE/SP release 2.1 or later.

Use in a Procedure

Return Codes:

OK	Successful.
NF	MEMD sublibrary not found.
SV	LIBR error response.

Example

List the locations in BIMLIB of all PHASEs that begin with BIMED:

```
=> libds bimed*.phase lib=bimlib
```

Output from LIBDS

```
=>
DISP  -> libds bimed*.phase lib=bimlib          SESS=A 1( 1) LINE=      0(   25)
-----1-----2-----3-----4-----5-----6-----7--
*==== -- TOP OF DISPLAY --
*==== libds bimed*.phase lib=bimlib
*==== RESULT OF SEARCH
*====                                     DATE: 2001-06-11
*====                                     TIME: 09:09
*====
*==== M E M B E R
*==== NAME      TYPE      LIBRARY  SUBLIB      CHAIN      CREATION    LAST
*====          |         |         |         |         |         DATE      UPDATE
*====          |         |         |         |         |         -----
*====
*==== BIMEDIT  PHASE      BIMLIB   ED          |         2001-04-19  - -
*==== BIMEDIT  PHASE      BIMLIB   EDTEMP      |         2001-04-19  - -
*==== BIMEDIT  PHASE      BIMLIB   ED00AM      |         2001-04-19  - -
*==== BIMEDIT  PHASE      BIMLIB   ED54AD      |         2001-04-19  - -
*==== BIMEDIT  PHASE      BIMLIB   ED54ADST     |         2001-04-19  - -
*==== BIMEDIT  PHASE      BIMLIB   ED54AMST     |         2001-04-19  - -
*==== BIMEDIT  PHASE      BIMLIB   QA           |         2001-04-19  - -
*==== BIMEDIT  PHASE      BIMLIB   EDPD2        |         2001-05-29  - -
*====
*==== BIMEDLUB PHASE      BIMLIB   P           |         2001-04-19  - -
*====
*==== BIMEDLUF PHASE      BIMLIB   T           |         2001-04-19  - -
*====
*==== BIMEDRB  PHASE      BIMLIB   T           |         2001-04-19  - -
*====
*==== BIMEDTMD PHASE      BIMLIB   P           |         2001-04-19  - -
*====
*==== -- END OF DISPLAY --
```


LIBRARYD (VSE version)

Use LIBRARYD to create a session displaying a list of members in a VSE sublibrary.

LIBRARYD can also be entered as LIBD.

Optional Operands	
MEMD	<p>the name and type to use to limit the members listed. The following forms are allowed:</p> <p>NAME . * members with name NAME and any type</p> <p>* . TYPE members with type TYPE and any name</p> <p>* . * all members</p> <p>NAME . TYPE the member NAME.TYPE, if present</p> <p>NAME* . * members whose names begin with NAME and any type</p> <p>* . TYPE* members whose types begin with TYPE and any name</p> <p>NAME* . TYPE* members whose names begin with NAME and whose types begin with TYPE.</p> <p>The currently attached VSE sublibrary (see ATTACHD) will be used unless the library and sublibrary are prefixed to the member name and type (i.e. library.sublibrary.member.type). If not specified, all members of the attached sublibrary are listed.</p> <p>When the OUT parameter is specified as STATUS the MEMD parameter will be interpreted as follows:</p> <p>LIB.SLIB a VSE directory status report for the specified library.sublibrary will be displayed.</p> <p>LIB.* a VSE directory status report for the specified library will be displayed.</p>
OUT	<p>specifies a library display option. OUT may be specified as one of the following values:</p> <p>NORMAL the library display will contain one line per member.</p> <p>FULL the library display will contain multiple lines per member (maximum amount of data).</p> <p>SHORT the library display will contain three members per line</p> <p>STATUS the library display contains status information about sublibrary.</p> <p>If OUT is not specified, the default is NORMAL.</p>

The following LCA commands used in conjunction with the LIBRARYD display can be quite useful:

E EDITD
L LISTD
P PURGED

See Chapter 5, LCA Commands, in the BIM-EDIT User Reference Manual for more information.

The LIBRARYD display has control information to the right of column 135 for use by LCA commands. If you use LIBRARYD display lines in another context, you may need to remove the control information using the BLANK or KEEP commands.

When Valid

BIM-EDIT must be running in its own partition and under VSE/SP release 2.1 or later. The user must have EXEC access level for the MEMD library.

Use in a Procedure

Return Codes:

OK Successful.
NF MEMD sublibrary not found.
SC Inadequate access level.

Example

List the OBJ type members of the current sublibrary:

```
=> libd *.obj
```

Output from LIBRARYD

```
=>
DISP -> libd *.obj                      SESS=A 2( 2) LINE=    0(    9)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
***** -- TOP OF DISPLAY --
***** DIRECTORY DISPLAY                SUBLIBRARY=BIMLIB.ED00AD      DATE: 96-03-13
*****                                     TIME: 07:03
*****
***** M E M B E R      CREATION  LAST      BYTES    LIBR CONT SVA  A- R-
***** NAME          TYPE    DATE      UPDATE  RECORDS  BLKS STOR ELIG MODE
*****
***** BIMBTTR  OBJ      94-07-15 95-09-06    12 R      1 YES  -  -  -
***** BIMBTWT  OBJ      94-07-15 95-09-06    37 R      4 YES  -  -  -
***** BIMBTXP  OBJ      94-07-15 95-09-06    78 R      7 YES  -  -  -
***** -- END OF DISPLAY --
```

LIBRARYD (MVS version)

Use LIBRARYD to create a session displaying a list of members in a Partitioned Data Set (PDS).

LIBRARYD can also be entered as LIBD.

Optional Operands	
MEMD	is the initial characters of the members to be listed. Members in the currently attached PDS (see ATTACHD) will be listed unless the PDS name is prefixed to the member name (i.e. pdsname.member). If not specified, all members of the attached PDS are listed.

The following LCA commands used in conjunction with the LIBRARYD display can be quite useful:

E EDITD
L LISTD
P PURGED

See Chapter 5, LCA Commands, in the BIM-EDIT User Reference Manual for more information.

The LIBRARYD display has control information to the right of column 135 for use by LCA commands. If you use LIBRARYD display lines in another context, you may need to remove the control information using the BLANK or KEEP commands.

When Valid

The user must have EXEC access level for the MEMD library.

Use in a Procedure

Return Codes:

OK Successful.
NF MEMD PDS not found.
SC Inadequate access level.

Examples

List the members whose names begin with BIM in the current PDS:

```
=> libraryd bim
```

Output from LIBRARYD

```

=>
DISP  -> libraryd bim                      SESS=A 1( 1) LINE=      0( 272)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==== -- TOP OF DISPLAY --
*==== PDS LIBRARY=BIM001.SRCLIB
*====
*==== MEMBER  VV.MM  DATE      LAST UPDATED  SIZE  INIT  MOD   USER
*====          CREATED      DATE      TIME      SIZE
*====
*==== BIMCOMND 01.13 03/10/95 03/24/95 13:37    64    17    55  WBYBXM
*==== BIMCSTAT 01.00 03/24/95 03/24/95 15:47   257   257    0  WBYBXM
*==== BIMCXASM 01.10 03/10/94 03/24/94 15:52    31    18   31  WBYBXM
*==== BIMCXCI7 01.01 03/11/94 03/11/94 18:02    63    63    1  WBYBXM
*==== BIMCXDBM 01.00 10/25/95 10/25/95 19:27     3     3    0  WBYBXM
*==== BIMCXLOD 01.10 10/03/94 01/14/95 08:47    13    21    0  WBYBXM
*==== BIMCXPRT 01.04 03/24/95 03/25/95 12:44    45    55   12  WBYBXM
*==== BIMCXULD 01.07 03/24/95 03/25/95 15:26    55    55   12  WBYBXM
*==== BIMEDIT  01.17 04/05/94 07/22/94 19:47     9     5    7  BIM001
*==== BIMPRDOS 01.03 05/01/94 06/18/94 11:39    19   13    0  WBYBXM
*==== BIMTEXTP 01.01 02/25/94 02/25/94 21:52    10   24    0  WBYBXM

```

LIBRARYL

Use LIBRARYL to display a list of the libraries you have access to. All libraries can be listed or only those starting with specified characters. When LIBRARYL is invoked online, it creates a DISP session for the results.

LIBRARYL may also be entered as LIBL.

Optional Operands	
LIB	Only those libraries starting with the LIB characters will be listed. If LIB is not specified, all libraries to which you have access will be listed. LIB can contain * and ? characters, which will match as described below in the Extended Search Pattern rules.
TITLE	If TITLE is specified, only libraries that contain its value in their TITLES are displayed. If the value of TITLE begins with a backslash character (\), matching will use Extended Search Pattern rules, as described below.

The LCA Y command in conjunction with the LIBRARYL display can be used to ATTACH to a library. The LCA L command can be used to display the directories of libraries that appear in the LIBRARYL display. See Chapter 5, LCA commands, in the BIM-EDIT User Reference Manual for more information.

The USER field is the value specified when the library was defined. SEC shows the access level for the user making the LIBL request. The date created is now displayed beyond column 72 and may be viewed with the alternate screen size if the line length is greater than 80 or with the VIEW command by shifting the screen to at least +10.

The LIBRARYL display has control information to the right of column 135 for use by LCA commands. If you use LIBRARYL display lines in another context, you may need to remove the control information using the BLANK or KEEP commands.

LIBRARYL operands can be treated as Extended Search Patterns, in which certain characters have special meanings:

- ? matches any single character
- * matches any sequence of zero or more characters
- @ matches zero or more instances of the character that follows the @
- ! matches the beginning of the operand
- < matches the beginning of the operand or any non-alphanumeric character
- > matches the end of the operand or any non-alphanumeric character
- | separates two patterns, EITHER of which may match for a successful search
- + separates two patterns, BOTH of which must match for a successful search
- ~ (NOT) reverses line-by-line success/failure of the pattern following it
- \ suppresses the special meaning of the character that follows the \

See Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual for more information about Extended Search Patterns.

Use in a Procedure

LIBRARYL normally returns OK. (See SIBRETCDD.)

If LIBRARYL is invoked in batch utility, the display destination is controlled by predefined variable SIBOUTPT.

Examples

List all accessible libraries:

```
=> libraryl
```

List all libraries starting with the characters "CRB":

```
=> libl crb
```

Output from LIBRARYL

```
=>
DISP  -> libl crb                                SESS=A 1( 1) LINE=    0(    8)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----
*====* -- TOP OF DISPLAY --
*====* -----
*====* LIBRARY                                TITLE                                USER    SEC
*====* -----
*====* CRBWORK                                WORK LIBRARY                                $SYS     DEF
*====* CRBDFH                                CICS TABLES (PCT, FCT, ETC.)                GJB     LIST
*====* CRBED51A                                BIM-EDIT 5.1A, SOURCE                          PG      DEF
*====* CRBED50                                BIM-EDIT 5.0, SOURCE                          $SYS     LIST
*====* -- END OF DISPLAY --
```

LIBRARYP (VSE version)

Use LIBRARYP to display a list of POWER job entries. Operands are provided to limit the entries that will be listed. When LIBRARYP is invoked online, it creates a DISP session for the results.

LIBRARYP may also be entered as LIBP or DQ.

Optional Operands	
QUEUE	is the POWER queue. Specify "RDR" to show only job entries from the RDR queue, "PUN" to show only job entries from the PUN queue, "LST" to show only job entries from the LST queue, or "XMT" to show only job entries from the XMT queue. If QUEUE is not specified, or if it is specified as "ALL", entries from all queues will be shown.
NAME	is the job name. If name is specified as "ALL", or is not specified, all jobs you have access to will be shown. If name is specified as "FREE", all jobs you have access to with a DISP=D or DISP=K will be shown. All other names will be assumed to be generic. The name can start with an asterisk (*) to be compatible with POWER's syntax, but it is not required.
NUMBER	is the job number. If NUMBER is specified, only entries whose job number matches NUMBER will be shown.
CLASS	is the job class. If CLASS is specified, only entries whose class matches CLASS will be shown.
USER	is site-defined. USER is up to 16 characters in length. The exit routine BIXPWQA can use this value to limit access.

For computability with POWER syntax, the following syntax exception is supported. If QUEUE starts with an asterisk, and NAME is not specified, the value in QUEUE will be interpreted as a generic name, and all queues will be searched. For example:

```
=> dq *ndj
```

will list job entries in all queues whose job names begin with NDJ.

DISP sessions such as that produced by the LIBRARYP command can be updated or "refreshed" by the REFRESH command. (You'll probably find it convenient to assign the REFRESH command to a PF key.)

The following LCA commands used in conjunction with the LIBRARYP display can be quite useful:

H	HOLD	R	RELEASE
L	LISTP	Q	FALTERP
P	PURGE		

See Chapter 5, LCA Commands, in the BIM-EDIT User Reference Manual for more information.

The LIBRARYP display is 130 columns wide, with the most commonly needed information in columns 1-80. Use the VIEW command, the SCREEN ALT option, or the SHIFT and PROPAGAT commands to access the additional data. The LIBRARYP display has control information beyond column 135 for use by LCA commands. If you use LIBRARYP display lines in another context, you may need to remove the control information using the BLANK or KEEP commands.

When Valid

Check with your System Administrator to determine the restrictions on access to POWER job entries at your site. The exit routine distributed with BIM-EDIT provides several restriction schemes your System Administrator can select among. (Use of and changes to the BIXPWQA exit routine are described in the System Reference Manual.)

Use in a Procedure

LIBRARYP normally returns OK. (See SIBRETC D.)

If LIBRARYP is invoked in batch utility, the display destination is controlled by predefined variable SIBOUTPT.

Examples

Display all entries in the list queue:

```
=> libraryp lst,all
```

Display all jobs with a class of 5:

```
=> libraryp rdr,class=5
```

Display all entries in all queues whose name starts with "T":

```
=> libp *t
```

Output from LIBRARYP

```
=>
DISP -> libp *t                                SESS=A 3( 3) LINE=    0(    8)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF DISPLAY --
*====*
*====* ----- POWER JOB ----- P D C S   CARDS/  CPY FORM  ORIGIN/  USER INFO
*====* QUE   NAME  NUMBER SG   - - - -   PAGES          DEST
*====*
*====* RDR TPRINT    225    3 L C           8              BM01
*====* LST TPRINT    27    3 H A          66  1              BM01
*====* LST TPRINT    28    3 H A         102  1              BM01
*====* LST TPRINT   226    3 H A           9  1              BM01
*====*
*====* -- END OF DISPLAY --
```


LIBRARYP (MVS version)

Use LIBRARYP to display a list of JES jobs and data sets. Operands are provided to limit the entries that will be listed. When LIBRARYP is invoked online, it creates a DISP session for the results.

LIBRARYP may also be entered as LIBP or DQ.

Optional Operands	
QUEUE	<p>specifies what type of data sets to list. Specify a one letter queue type:</p> <p>H Specifies hold queue entries</p> <p>I Specifies input queue entries</p> <p>O Specifies output queue entries</p> <p>A Specifies input queue, held queue and output queue entries</p> <p>If QUEUE is not specified, A is assumed.</p>
USER	<p>is site-defined. USER can be up to 16 characters in length. The exit routine BIXPWQA is provided addressability to this value.</p>

All jobs available to you are listed.

Data sets for each job are consolidated on the display. In "hold queue", a new line is displayed when class changes. In "output queue", a new line is displayed when outgrp changes. "Input queue" entries exist only if the job has not completed processing. If you need the ddnames or data set numbers, use INQUIREP.

DISP sessions such as that produced by the LIBRARYP command can be updated or "refreshed" by the REFRESH command. (You'll probably find it convenient to assign the REFRESH command to a PF key.)

The following LCA commands used in conjunction with the LIBRARYP display can be quite useful:

H	HOLD	R	RELEASE
L	LISTP	Q	FALTERP
P	PURGE		

See Chapter 5, LCA Commands, in the BIM-EDIT User Reference Manual for more information.

The LIBRARYP display has control information to the right of column 135 for use by LCA commands. If you use LIBRARYP display lines in another context, you may need to remove the control information using the BLANK or KEEP commands.

When Valid

Check with your System Administrator to determine the restrictions on access to JES data sets at your site. The exit routine distributed with BIM-EDIT provides several restriction schemes your System Administrator can select among. (Use of and changes to the BIXPWQA exit routine are described in the System Reference Manual.)

Use in a Procedure

LIBRARYP normally returns OK. (See SIBRETCO.)

If LIBRARYP is invoked in batch utility, the display destination is controlled by predefined variable SIBOUTPT.

Examples

Display all entries in the hold queue:

```
=> libraryp h
```

Display all entries in the queues:

```
=> libp
```

Output from LIBRARYP

See LIBPOUT for format options.

```

M>
DISP -> libraryp                                SESS= 1( 1) LINE= 0( 20)
-----1-----2-----3-----4-----5-----6-----7-----
*==== -- TOP OF DISPLAY --
*==== ----- INPUT QUEUE -----
*==== JOB ---- T C  ORIG  PR EXEC STAT  SUBMIT  JOBCARD
*==== NAME  NUMBER Y L  NODE  TY SYS  -US  USERID  USERID
*====
*==== MVSEDT  1106 J D ?                5 MP3K      FKE      PG
*==== ----- HELD QUEUE -----
*==== JOB ----- C  GROUP  FORMS  FCB  CREATE  LINES  MAX
*==== NAME  NUMBER L  NAM/QUAL/CPY  ID      DATE
*====
*==== PORTMAP  STC00919 K 1.1.1      STD      ****  04/30/2003      2 CC 40
*==== PORTMAP  STC00919 K 2.1.1      STD      ****  04/30/2003      1 CC 40
*==== IRRDPTAB STC00915 K 1.1.1      STD      ****  04/17/2003      1 CC 00
*==== BCICS53  JOB00869 D 1.1.1      STD      ****  04/17/2003     497 CC 00
*==== BCICS53  JOB08493 D 1.1.1      STD      ****  01/27/2003     544 CC 00
*==== ----- OUTPUT QUEUE -----
*==== JOB ----- C  GROUP  FORMS  FCB  CREATE  LINES  MAX
*==== NAME  NUMBER L  NAM/QUAL/CPY  ID      DATE
*====
*==== BCICS53  JOB01101 D 2.1.1      STD      ****  04/30/2003     200 CC 00
*==== SDSF     STC00479 A 1.1.1      STD      ****  03/17/2003     223 CC 00
*==== -- END OF DISPLAY --

```

LIBRARYQ

Use LIBRARYQ to display a list of all messages in the incoming-mail, mail-set-aside, and outgoing-mail queues. When LIBRARYQ is invoked online, it creates a DISP session for the results.

LIBRARYQ may also be entered as LIBQ.

LIBRARYQ has no operands.

For each message, the following information is displayed:

- Source or destination user ID (as appropriate).
- Four digit number assigned to the message.
- Date/time the message was sent.
- Message title if specified, otherwise the first 40 message characters.

To display the text of a message, use the OPEN command. To purge a message, use DISCARD or PURGEQ. To send a message, use MAIL or MAILI.

The following LCA commands can be useful with the LIBRARYQ display:

```
L  OPEN
P  PURGEQ
```

See Chapter 5, LCA Commands, in the BIM-EDIT User Reference Manual for more information.

The LIBRARYQ display has control information to the right of column 135 for use by LCA commands. If you use LIBRARYQ display lines in another context, you may need to remove the control information using the BLANK or KEEP commands.

Use in a Procedure

LIBRARYQ normally returns OK. (See SIBRETC D.)

If LIBRARYQ is invoked in batch utility, the display destination is controlled by predefined variable SIBOUTPT.

Example

The following is an example of the LIBRARYQ output:

```
=>
DISP  -> libraryq                                SESS=A 1( 1) LINE=    0(    16)
-----1-----2-----3-----4-----5-----6-----7--
*==== -- TOP OF DISPLAY --
*====
*==== INCOMING MAIL
*==== FROM      ID      DATE      TIME      TITLE
*====-----
*==== AMS      1998 04/24/1995 10:40:15 DOCUMENTATION REQUEST
*==== AMS      1999 04/24/1995 11:15:05 Folders you requested have arrived.
*====-----
*==== MAIL SET ASIDE
*==== FROM      ID      DATE      TIME      TITLE
*====-----
*==== PG      1931 04/21/1995 15:02:06 Could you provide info concerning XPCC
*====-----
*==== OUTGOING MAIL
*==== TO        ID      DATE      TIME      TITLE
*====-----
*==== MB      1976 04/23/1995 12:58:53 Yes. Mailing is set.
*==== -- END OF DISPLAY --
```

LIBRARYU

Use LIBRARYU to create a session displaying a list of users.

LIBRARYU may also be entered as LIBU.

Optional Operands	
USER	Only those users which start with USER characters will be listed. If USER is not specified, all users will be listed. USER can contain * and ? characters, which will match as described below in the Extended Search Pattern rules.
NAME	If NAME is specified, only users whose name contains its value are displayed. If the value of NAME begins with a backslash character (\), matching will use Extended Search Pattern rules, as described below.

The LIBRARYU display normally shows only the user ID, the user name, and the logon terminal id. If the user issuing the LIBRARYU command has an OPER or ADM command security level, additional information is displayed.

The following LCA commands used in conjunction with the LIBRARYU display can be quite useful:

P PURGEU
Q FALTERU

See Chapter 5, LCA Commands, in the BIM-EDIT User Reference Manual for more information.

The LIBRARYU display has control information to the right of column 135 for use by LCA commands. If you use LIBRARYU display lines in another context, you may need to remove the control information using the BLANK or KEEP commands.

LIBRARYU operands can be treated as Extended Search Patterns, in which certain characters have special meanings:

- ? matches any single character
- * matches any sequence of zero or more characters
- @ matches zero or more instances of the character that follows the @
- ! matches the beginning of the operand
- < matches the beginning of the operand or any non-alphanumeric character
- > matches the end of the operand or any non-alphanumeric character
- | separates two patterns, EITHER of which may match for a successful search
- + separates two patterns, BOTH of which must match for a successful search
- ~ (NOT) reverses line-by-line success/failure of the pattern following it
- \ suppresses the special meaning of the character that follows the \

See Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual for more information about Extended Search Patterns.

Use in a Procedure

LIBRARYU normally returns OK. (See SIBRETCD.)

Example

List all users whose name contains the word "atcheson"

```
=> libraryu name=\<atcheson>
```

Output of LIBRARYU Command

```
=>
DISP -> libraryu name=\<atcheson>          SESS=A 1( 1) LINE=    0(    5)
-----1-----2-----3-----4-----5-----6-----7--
*==* -- TOP OF DISPLAY --
*==* -----
*==* USER          NAME          TERMINAL
*==* -----
*==* ADR      ATCHESON, DEAN      WA02
*==* AJP      JANE ATCHESON      -----
*==* -- END OF DISPLAY --
```

Output of LIBRARYU Command by userid with OPER or ADM security

```
=>
DISP -> libraryu name=\<atcheson>          SESS=A 1( 1) LINE=    0(    5)
-----1-----2-----3-----4-----5-----6-----7--
*==* -- TOP OF DISPLAY --
*==* -----
*==* USER          NAME          TERMINAL - SECURITY - USAGE  S
*==* -----
*==* ADR      ATCHESON, DEAN      WA02      CMD  LIB PWR  -----
*==* AJP      JANE ATCHESON      -----  OPER DEFS 3    2500  0
*==* -- END OF DISPLAY --
```

USAGE is the number of times ENTER or a PF key has been pressed since the last time an ALTERU USAGE=CLEAR was entered.

SS is the number of active sessions.

LIBSDL (VSE only)

Use LIBSDL to create a session displaying the output of a VSE LIBR LISTDIR SDL command.

The operands for the LIBSDL command are those supported by the LIBR LISTDIR SDL command. BIM-EDIT does not edit them. All operands specified following the LIBSDL command are passed directly to LIBR.

The syntax of the LISTDIR SDL command is summarized here, but can also be found in the VSE/ESA System Control Statements manual.

The following command formats are supported:

- `Libsdl`
- `Libsdl phase=mn`
- `Libsdl phase=mn*`
- `LIBsdl phase=mn.mt`

where:

<code>mn</code>	the member name to be listed.
<code>mt</code>	the member type to be listed.

If the `phase=` operand is omitted, all entries in the SDL will be listed.

The LCA command 'L' can be used on an entry in the SDL to cause the BIM-EDIT CORE command to be invoked to display the contains of the entry.

The LIBSDL display has control information to the right of column 135 for use by LCA commands. If you use LIBDS display lines in another context, you may need to remove the control information using the BLANK or KEEP commands.

When Valid

BIM-EDIT must be running in its own partition and under VSE/SP release 2.1 or later.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Member not found.
SV	LIBR error response.

Example

List the contents of the SDL:

```
=> libsd1
```

Output from LIBSDL

```
=>
DISP -> libsd1                                SESS=A 1( 1) LINE=      0( 448)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----
*==== -- TOP OF DISPLAY --
*==== libsd1
*==== STATUS DISPLAY          SDL AND SVA                      DATE: 2001-07-17
*====                                                                TIME: 09:53
*====
*==== SDL          TOTAL ENTRIES :    452  (100%)
*====              USED ENTRIES :    423  ( 94%)
*====              FREE ENTRIES :     29  (  6%)
*====
*==== SVA(24)      TOTAL SPACE   :   2840K (100%)
*====              USED SPACE   :   1950K ( 69%)
*====              - PFXED AREA:   123K  (  4%)  START AT: 0037F220
*====              FREE SPACE   :    890K ( 31%)
*====
*==== SVA(31)      TOTAL SPACE   :   3512K (100%)
*====              USED SPACE   :   3063K ( 87%)
*====              - PFXED AREA:   550K  ( 16%)  START AT: 026E45B8
*====              FREE SPACE   :    449K ( 13%)
*====
*====
*==== DIRECTORY DISPLAY  SDL                      DATE: 2001-07-17
*====                                                                TIME: 09:53
*====
*==== M E M B E R  ORIGIN SVA/MOVE  LOADED  PHASE  ADDRESS  ENTRY POINT
*==== NAME        TYPE  SYSLIB    MODE   INTO SVA  SIZE    IN SVA   IN SVA
*====
*==== $$BACLOS PHASE  YES    MOVE    31      562  02483028 02483028
*==== $$BATNA PHASE  YES    MOVE    31     1256 02483260 02483260
*==== $$BATNB PHASE  YES    MOVE    31      718 02483748 02483748
*==== $$BATNK PHASE  YES    MOVE    31     1104 02483A18 02483A18
*==== $$BATNR PHASE  YES    MOVE    31      389 02483E68 02483E68
*==== $$BCEOV1 PHASE YES    NO      NO      74    -      -
*==== $$BCLOSE PHASE YES    MOVE    31     1192 02483FF0 02483FF0
*==== $$BCLOS2 PHASE YES    MOVE    31      624 02484498 02484498
*==== $$BCLOS5 PHASE YES    MOVE    31     1032 02484708 02484708
*==== $$BCLRPS PHASE YES    MOVE    31      720 02484B10 02484B10
*==== $$BCVSAM PHASE YES    MOVE    31      768 02484DE0 02484DE0
```


LIST

Use LIST to create a LIST session of an existing member, \$LOG, \$MAIL, or \$STACK.

LIST may also be entered as LI.

Optional Operands	
MEM	is an existing member, \$LOG, \$MAIL, or \$STACK. \$LOG is the log area, \$MAIL the mail log, and \$STACK is the stack area. If MEM is not specified, the "last referenced member" is used.
PSWD	If the member is password protected, the password must be entered. The password may be up to 8 characters in length consisting of any characters.

LIST can also be invoked by entering the L command in the LCA area of a LIBRARY display.

LIST sessions do not allow updates to take place.

If the member is currently being edited, LIST will display the text of the member ignoring any changes made in edit session.

LIST sets "last referenced member".

When Valid

The user must have LIST access level for the MEM library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Member not found.
SC	Inadequate access level.

LIST sets the TXM variables to the attributes of MEM.

Examples

List member RM3521:

```
=> list rm3521
```

List the log:

```
=> list $log
```

List the mail log:

```
=> li $mail
```

List the stack area:

```
=> list $stack
```

LISTD (VSE version)

Use LISTD to create a LIST session of a VSE sublibrary member.

LISTD may also be entered as LID.

Optional Operands	
MEMD	the name and type of an existing VSE sublibrary member. The currently attached sublibrary (see ATTACHD) will be used unless the library and sublibrary are prefixed to the member name and type (i.e. library.sublib.member.type). If MEMD is not specified, the "last referenced VSE sublibrary member" will be used.

The initial characteristics of the LISTD session are those established by the System Administrator for the member type. This is implemented by copying the attributes of a "template member" in the BIM-EDIT Site or System Model library. The template member used is the one whose name matches the member type.

LISTD sets the "last referenced VSE sublibrary member".

See the HEX option of the SCREEN command for working with object members.

When Valid

BIM-EDIT must be running in its own partition and under VSE/SP release 2.1 or later. The user must have LIST access level for the MEMD library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Member not found.
SC	Inadequate access level.

Examples

Create a list session of member DAVE1.PHASE in sublibrary PERSONAL.DAVE:

```
=> listd personal.dave.dave1.phase
```

Create a list session of member JMY8806.C in the current sublibrary:

```
=> lid jmy8806.c
```

Output of LISTD Command

```
=>
LISTD BIMLIB.T.JMY8806.C                      SESS=A 2( 2) LINE=      0(   48)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF MEMBER --
*====*      *PROPOSED EMPLOYEE FILE RECORD STRUCTURE 6/88 JMY
*====*      01  EMPLOYEE-REC.
*====*          03  EMP-NAME.
*====*              05  EMP-FIRST-NAME      PICTURE X(20) .
*====*              05  EMP-LAST-NAME      PICTURE X(20) .
*====*          03  EMP-HIRE-DATE.
*====*              05  EMP-HIRE-MM        PICTURE 99 .
*====*              05  EMP-HIRE-DD        PICTURE 99 .
*====*              05  EMP-HIRE-YY        PICTURE 99 .
```

LISTD (MVS version)

Use LISTD to create a LIST session of a Partitioned Data Set (PDS) member.

LISTD may also be entered as LID.

Optional Operands	
MEMD	the name of the PDS member. The currently attached PDS (see ATTACHD) will be used unless the PDS name is prefixed to the member name, that is, pdsname.member or pdsname(member). If MEMD is not specified, the "last referenced PDS member" will be used.

The initial characteristics of the LISTD session are those established by the System Administrator for the member type. This is implemented by copying the attributes of a "template member" in the BIM-EDIT Site or System Model library. The template member used is the one whose name matches the last qualifier in the PDS name.

LISTD sets the "last referenced PDS member".

LISTD is allowed only for fixed block members with a record length 253 or less. Members containing source code generally meet this qualification, but load modules do not.

When Valid

The user must have LIST access level for the MEMD library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Member not found.
SC	Inadequate access level.

Examples

Create a list session of member DAVE1 in PDS PERSONAL.DAVE:

```
=> listd personal.dave.dave1
```

Create a list session of member JMY8806 in the current sublibrary:

```
=> lid jmy8806
```

Output of LISTD Command

```
=>
LISTD BIMLIB.JMY8806                      SESS=A 2( 2) LINE=      0(   48)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*  -- TOP OF MEMBER --
*====*          *PROPOSED EMPLOYEE FILE RECORD STRUCTURE 6/88 JMY
*====*          01  EMPLOYEE-REC.
*====*              03  EMP-NAME.
*====*                  05  EMP-FIRST-NAME          PICTURE X(20) .
*====*                  05  EMP-LAST-NAME           PICTURE X(20) .
*====*              03  EMP-HIRE-DATE.
*====*                  05  EMP-HIRE-MM              PICTURE 99 .
*====*                  05  EMP-HIRE-DD              PICTURE 99 .
*====*                  05  EMP-HIRE-YY              PICTURE 99 .
```

LISTP (VSE version)

Use LISTP to create a LIST session of a POWER job entry or of the output of a currently executing VSE job.

LISTP may also be entered as LP.

Optional Operands	
PRM1-4	<p>select the POWER job entry or currently executing output to be listed.</p> <p>The four operands can be specified in one of two general formats:</p> <p>For a job entry: queue, job name, job number, segment</p> <p>For executing output: part, cuu</p> <p>For a job entry, the four operands allow you to specify the queue (RDR,LST,PUN,XMT), the job name, the job number, and the segment number. They can be specified in any order, except that segment, if specified, must follow job number. The following indicates which entry will be selected based on the operands specified:</p> <p>queue, job name, job number, segment</p> <p> The specified queue will be searched for the entry for the specified name, number, and segment.</p> <p>queue, job name</p> <p> The specified queue will be searched for the first entry for the specified name.</p> <p>queue, job number</p> <p> The specified queue will be searched for the first entry for the specified number, without regard to the job name.</p> <p>job number</p> <p>job name</p> <p> The LST queue will be searched for the first entry for the specified number or name.</p> <p>queue</p> <p> The specified queue will be searched for the first entry for the job name and number of the "last referenced POWER job entry"</p>

Optional Operands (continued)	
	<p>For executing output, PRM1-2 specify the partition and CUU of the output to be listed:</p> <p>Part, cuu</p> <p>'part' is the 2-character partition id of the executing job, and cuu is the 3-character cuu of the LST or PUN queue entry currently being generated.</p> <p>If 'cuu' is omitted, the first LST queue entry for the specified 'part' will be displayed.</p> <p>'part' must be specified as the first operand.</p> <p>If PRM1-4 is not specified at all, the "last referenced POWER job entry" will be used.</p>

LISTP can also be invoked by entering the L command in the LCA area of a LIBRARYP display.

Initially, the attributes of a LISTP session are ZONE=1-253 for LOCATE, LOCATEU, and QUALIFY; and FCOL=1 for FIND, FINDUP, NFIND, and NFINDUP. These can be changed with the SESS command. Hexadecimal display is initially off. This can be changed with the SCREEN HEX command.

On 80 column terminals, the VIEW command can be used to see all columns of a wide POWER job entry. You may wish to set PF keys to various VIEW commands.

If a POWER job entry is deleted while you are listing it, your session will collapse to a one-line message indicating the entry has been deleted. You must terminate this session with END or SAVE.

When the LISTP command is being used to display the output of a currently executing job, the number of lines in the session will be continuously updated to reflect the actual output. You can use the BOTTOM command to move to the last line generated as of the time the BOTTOM command is entered. However, since the session line count is being updated continuously, the BOTTOM may not always be the actual bottom by the time the screen is generated.

The INQUIREP command can be used to display a list of all currently executing output being generated in your VSE system.

LISTP sets the "last referenced POWER job entry".

When Valid

Check with your System Administrator to determine the restrictions on access to POWER job entries at your site. The exit routine distributed with BIM-EDIT provides several restriction schemes your System Administrator can select among. (Use of and changes to the BIXPWQA exit routine are described in the System Reference Manual.)

Use in a Procedure

Return Codes:

OK	Successful.
NF	Job entry not found.
*	Codes as defined by the BIXPWQA exit routine.

LISTP sets the PWR variables to the attributes of the POWER job entry listed.

Examples

Create a LISTP session for the job entry XM3050 in the RDR queue:

```
=> listp rdr,xm3050
```

Create a LISTP session for segment 2 of job entry 1278 in the PUN queue:

```
=> lp pun,1278,2
```

Create a LISTP session for the job entry ACPAY805 in the LST queue:

```
=> listp acpay805
```

Result of LISTP Command

```
=>
LISTP LST,ACPAY805,2988                      SESS=A 2( 2) LINE=      0( 4038)
-----1-----2-----3-----4-----5-----6-----7--
*==*  -- TOP OF JOB --
*==*
*==*                                SAMUELSON INSURANCE, INC.
*==*
*==*                                ACCOUNTS PAYABLE 05/01/95 - 06/01/95
*==*
*==*                                CHECK
*==*      DATE      NMBR  PAYEE              ACCT DESCRIPTION              AM
*==*
*==* 05/08/95  DEP  DEPOSIT              0300 ACME SOFT BROKERS, FEB          398
*==*                                         DEPOSIT              TOTAL
*==*
*==* 05/08/95 30811 DAWN SAMUELSON        1001 NET WAGES, APRIL 1995          892
*==*                                         CHECK 30811 TOTAL
```

Create a LISTP session for the output being generated to X'02E' by the job currently executing in partition FA:

```
=> listp fa,02e
```



```
=>
LISTP FA,02E,BIJEDITT,9580                      SESS=A 2( 2) LINE=      0    24)
-----1-----2-----3-----4-----5-----6-----7-----8
-- TOP OF JOB --

SYSTEM=BIMEDITT
BIRSIXPA  INITIALIZING XPCC INTERFACE
BIRSIOPA  OPENING DATABASE
BIRSIQAA  OPENING POWER INTERFACE
BIRSIDCA  INITIALIZING CONSOLE HARDCOPY SUPPORT
BIRSIMGA  OPENING IESMSGs DATASET
BIRSIMPA  MAPPING MEMORY
BIRSIIRA  PERFORMING SYSTEM RECOVERY
BIRSIDKA  BUILDING DISK TABLE, KEY SEQUENCE INDEX
BIRSIMSA  BUILDING MESSAGE TABLE
BIRSIIRA  PERFORMING RECOVERY FOR USER "GOP"
BIRXLSIA  BUILDING SITE COMMAND LIST
BIRSIABA  INITIALIZING SUBMISSION INTERFACE
BIRSIIRA  INITIALIZING LIBR INTERFACE
BIRSIABA  INITIALIZING SITE STANDARD VARIABLES
BIRSIIRA  INITIALIZING VTAM INTERFACE
BIRSIIRA  INITIALIZING TCP/IP INTERFACE
BIMEDITA  SYSTEM "BIMEDITT" INITIALIZATION COMPLETE

BIRPWGCA  ACTIVATING POWER COMPLETION MESSAGE INTERFACE
BIRPWNYA  ACTIVATING POWER JOB NOTIFY INTERFACE
BIRCNCIA  CONSOLE ACCESS ROUTINE ACTIVATING
-- END OF JOB --
```

LISTP (MVS version)

Use LISTP to create a LIST session of JES data sets.

LISTP may also be entered as LP.

Optional Operands	
QUEUE	specifies the JES queue. QUEUE may be specified as "I" for the input queue, "H" for the held queue or "O" for the output queue.
JOB	selects the JES job to be displayed. If JOB contains letters, it specifies a jobname label from the Job Control Language JOB statement. If JOB is entirely digits, it specifies a job number assigned by JES. If not specified, the job from "last referenced JES data sets" will be used.
GROUP	selects the data sets to be displayed within the specified job. If GROUP is a single letter or digit, it specifies all data sets in the GROUP class in the JES "hold queue". If GROUP is "SYSOUT", it specifies all SYSOUT data sets. If GROUP is any other multiple character value, it specifies all data sets in the GROUP outgrp in the JES "output queue". An outgrp may be composed of three parts separated by periods in the form "name.qualifier.copy" where "name" is the JES or user-specified name, qualifier is a JES-assigned number, and copy is the copy number. All data sets matching a partial outgrp specification (for example, "name" only) will be displayed. To distinguish an outgrp with a one-character name from a class, follow the name with a period.
DSET	selects the data set to be displayed within the specified job. DSET can specify a user data set name from the ddname label of a Job Control Language DD statement. The data set name may be qualified by entering it in the form stepname.ddname. If the data set name is not unique within the job, the first instance will be used. You can specify JES pre-defined data sets, of which the most useful is \$JCL, which contains the Job Control Language for the job. DSET can also be specified as a JES-assigned data set number which is always unique. Use the INQUIREP command to display a list of data sets and their numbers. If none of JOB, GROUP and DSET are specified, the "last referenced data sets" are displayed. If only JOB is specified, all SYSOUT data sets are displayed.

LISTP can also be invoked by entering the L command in the LCA area of a LIBRARYP display.

Initially, the attributes of a LISTP session are ZONE=1-253 for LOCATE, LOCATEU, and QUALIFY; and FCOL=1 for FIND, FINDUP, NFIND, and NFINDUP. These can be

changed with the SESS command. Hexadecimal display is initially off. This can be changed with the SCREEN HEX command.

On 80 column terminals, the VIEW command can be used to see all columns of wide JES data sets. You may wish to set PF keys to various VIEW commands.

If JES data sets are deleted while you are listing them, your session will collapse to a one-line message indicating the entry has been deleted. You must END or DISCARD this session.

LISTP sets "last referenced JES data sets".

When Valid

Check with your System Administrator to determine the restrictions on access to JES data sets at your site. The exit routine distributed with BIM-EDIT provides several restriction schemes your System Administrator can select among. (Use of and changes to the BIXPWQA exit routine are described in the System Reference Manual.)

Use in a Procedure

Return Codes:

OK	Successful.
NF	JES job or data sets not found.
*	Codes as defined by the BIXPWQA exit routine.

LISTP sets the JCT, JQE, and PDB variables to the attributes of the JES data sets listed. If GROUP is an outgrp, the JOE variables are also set.

Examples

Create a LISTP session for the sysout data sets of job 6543:

```
=> listp 6543
```

Create a LISTP session for data set SYSPRINT in step GO of job XM3050:

```
=> lp xm0350,dset=go.sysprint
```

Create a LISTP session for data sets in hold queue class A in the last referenced job:

```
=> listp group=a
```

Result of LISTP Command

```
=>
LISTP U0052E/7928,A                      SESS=A 2( 2) LINE=      0( 4038)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*  -- TOP OF JOB  --
*====*
*====*                      SAMUELSON INSURANCE, INC.
*====*
*====*                      ACCOUNTS PAYABLE 05/01/95 - 06/01/95
*====*
*====*      CHECK
*====*      DATE      NMBR PAYEE      ACCT DESCRIPTION      AM
*====*
*====* 05/08/95  DEP  DEPOSIT      0300 ACME SOFT BROKERS, FEB      398
*====*                                DEPOSIT      TOTAL
*====*
*====* 05/08/95 30811 DAWN SAMUELSON 1001 NET WAGES, APRIL 1995      892
*====*                                CHECK 30811 TOTAL
*====*
*====* 05/08/95 30812 JOHN ANDREWS 1001 NET WAGES, APRIL 1995      528
*====*                                CHECK 30812 TOTAL
```

LOCATE

Use LOCATE to position the current session forward from the current line to the first line having a specified string of characters in a specified column range.

LOCATE may also be entered as LOC or L.

Optional Operands	
STR	is the string to search for. Up to 72 characters can be specified. If the value of STR begins with the backslash character (\), the search will use Extended Search Pattern rules, as described below. If STR is not specified, the value for STR and any other operands not specified will be inherited from previous search commands, also as described below.
ZONE	is the column range for the search. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to limit the search to columns 10 to 20, specify ZONE as "10-20". ZONE=5 is the same as ZONE=5-5, ZONE=5-* is the same as ZONE=5-253 and ZONE=-5 is the same as ZONE=1-5. If ZONE and STR are not specified, the ZONE specification is inherited from previous search commands, as described below. If ZONE, after any inheriting, is not specified, the current session zone is searched (see the ZONE operand of the DEFINE, ALTER, and SESS commands.)
CASE	specifies whether upper/lower case should be considered when determining matches. If specified as "U", case is ignored ("the" matches "The"). If specified as "M", both case and letters must match. If CASE and STR are not specified, the CASE specification is inherited from previous search commands, as described below. If CASE, after any inheriting, is not specified, "U" is assumed.

The search begins with the line after the current line and proceeds until a match is found. An error message will display if no match is found.

See also the SEARCH command.

After LOCATE, the CURSOR command will place the cursor at the match.

The repeat-command symbol (&) before the LOCATE command is useful to search, stopping at all lines matching STR. For example:

```
=> &locate xlib;cursor
```

LOCATE can be used with all session types. For example, LISTP sessions can be searched (for compilation error messages or other tell-tale results) and LIBRARY displays can be searched for words in member titles.

If your terminal is running under control of SNA, you may interrupt a long LOCATE command by pressing the ATTN key. If the value of STR begins with a backslash (\), BIM-EDIT treats STR as an Extended Search Pattern, with certain characters having special meanings:

- ? matches any single character
- * matches any sequence of zero or more characters
- @ matches zero or more instances of the character that follows the @
- ! matches the beginning of the zone
- < matches the beginning of the zone or any non-alphanumeric character
- > matches the end of the zone or any non-alphanumeric character
- | separates two patterns, EITHER of which may match for a successful search
- + separates two patterns, BOTH of which must match for a successful search
- ~ reverses line-by-line success/failure of the pattern following it
- % marks the position on the line where the match is considered to occur
- \ suppresses the special meaning of the character that follows the \
- # matches the first numeric character found
- ^ matches the first alphabetic character found

A common use of the extended search string is to position the cursor at the first non-blank character in a line. The following locate command will perform this function:

```
=> locate @"\@ %?";cursor
```

A common mistake is to use the following locate command:

```
=> locate "\~ ";cursor
```

This search will only position to text lines that do not contain any blanks because the search request says, in effect, find a blank anywhere in the search area, and then reverse the match to a false condition.

See Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual for more information about Extended Search Patterns.

If STR is not specified on the command, BIM-EDIT uses as default operand values those from previous search commands. Using this feature, you can easily request searches which are similar or identical to the last one. For example, after a LOCATE command which failed because the instance is backward from the current line, you need only enter

```
=> locateu
```

to repeat exactly the same command in the backward direction. To achieve this, BIM-EDIT stores a previous value for the STR, COL, CASE, and ZONE operands of the search commands (EXAMINE, FIND, FINDUP, LOCATE, LOCATEU, NFIND, NFINDUP, QUALIFY, SCAN or SEARCH). These previous values will be inherited by operands omitted from a search command which does not specify STR. Whenever a search command specifies STR, the previous values are set to those specified on that command or "not specified".

Use in a Procedure

Return Codes:

OK Successful.
 AT Command interrupted by ATTN key.
 NF String not found.

LOCATE sets the predefined variable SSDCOL1 to the column number where the match is found (if in split-screen mode, and if the lower logical screen is primary, SSDCOL2 is set instead).

Example

Screen before LOCATE command

```
=> locate iolength
LIST 4217.IOROUT                                SESS=A 1( 1) LINE= 34( 256)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==== MVC TXLBLKSZ,=H'4096' SET CONSTANT TXLFL RECORD SIZE
*==== L R1,PM4 GET BLOCK COUNT ADDRESS
*==== LH R1,0(R1) GET BLOCK COUNT
*==== MH R1,TXLBLKSZ COMPUTE NUMBER OF BYTES NEEDED
*==== STH R1,IOLNGTH SAVE IN I/O CONTROL FIELD
*==== L R1,PM2 GET EXTENT INFO PARM
*==== MVC IOEXTENT,0(R1) USE FILE EXTENT 1 INFO FOR RELOCATE
*==== BAL R11,RELOCATE RELOCATE CHANNEL PROGRAM
*==== BAL R11,CALCSEEK CALCULATE SEEK ADDRESS
*==== BAL R11,EXCPDISK DO I/O
*==== B RETURN RETURN TO CALLER
*==== *****
*==== * SUBROUTINES
*==== * 1) RMZPQAR - HANDLES INPUT FROM CARDS
*==== * 2) RMZPQXR - HANDLES INPUT FROM TAPE
*==== * 3) RMZPQXA - HANDLES INPUT FROM SCREEN
*==== * 4) RMZPQXM - HANDLES INPUT FROM MEMBER
*==== * 5) RMZPRS1 - PARSES INPUT STRING, TYPE 1
*==== * 6) RMZPRS2 - PARSES INPUT STRING, TYPE 2
```

Screen after LOCATE command

```
=>
LIST 4217.IOROUT                                SESS=A 1( 1) LINE= 38( 256)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==== STH R1,IOLNGTH SAVE IN I/O CONTROL FIELD
*==== L R1,PM2 GET EXTENT INFO PARM
*==== MVC IOEXTENT,0(R1) USE FILE EXTENT 1 INFO FOR RELOCATE
*==== BAL R11,RELOCATE RELOCATE CHANNEL PROGRAM
*==== BAL R11,CALCSEEK CALCULATE SEEK ADDRESS
*==== BAL R11,EXCPDISK DO I/O
*==== B RETURN RETURN TO CALLER
*==== *****
*==== * SUBROUTINES
*==== * 1) RMZPQAR - HANDLES INPUT FROM CARDS
*==== * 2) RMZPQXR - HANDLES INPUT FROM TAPE
*==== * 3) RMZPQXA - HANDLES INPUT FROM SCREEN
*==== * 4) RMZPQXM - HANDLES INPUT FROM MEMBER
*==== * 5) RMZPRS1 - PARSES INPUT STRING, TYPE 1
*==== * 6) RMZPRS2 - PARSES INPUT STRING, TYPE 2
*==== * 7) RMZPRS3 - PARSES INPUT STRING, TYPE 3
*==== *
*==== * ALL SUBROUTINES RECEIVED ADDRESSABILITY TO
*==== * EIB - EXEC INTERFACE BLOCK
```

LOCATEU

Use LOCATEU to position the current session backward from the current line to the first line having a specified string of characters in a specified column range.

LOCATEU may also be entered as LOCUP, LOCU, LUP, or LU.

Optional Operands	
STR	is the string to search for. Up to 72 characters may be specified. If the value of STR begins with the backslash character (\), the search will use Extended Search Pattern rules, as described below. If STR is not specified, the value for STR and any other operands not specified will be inherited from previous search commands, also as described below.
ZONE	is the column range for the search. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to limit the search to columns 10 to 20, specify ZONE as "10-20". ZONE=5 is the same as ZONE=5-5, ZONE=5-* is the same as ZONE=5-253 and ZONE=-5 is the same as ZONE=1-5. If ZONE and STR are not specified, the ZONE specification is inherited from previous search commands, as described below. If ZONE, after any inheriting, is not specified, the current session zone is searched (see the ZONE operand of the DEFINE, ALTER, and SESS commands.)
CASE	specifies whether upper/lower case should be considered when determining matches. If specified as "U", case is ignored ("the" matches "The"). If specified as "M", both case and letters must match. If CASE and STR are not specified, the CASE specification is inherited from previous search commands, as described below. If CASE, after any inheriting, is not specified, "U" is assumed.

The search begins with the line preceding the current line and proceeds upward until a match is found. An error message will display if no match is found. The line containing the match will become the current line of the screen. (Since it is not possible to position AFTER the last line in a session, any match for STR on the last line will be ignored by LOCATEU.)

After LOCATEU, the CURSOR command will place the cursor at the match.

The repeat-command symbol (&) before the LOCATEU command provides a useful way to search, stopping at all lines containing STR. For example:

```
=> &locateu xlib;cursor
```

If your terminal is running under control of SNA, you may interrupt a long LOCATEU command by pressing the ATTN key. If the value of STR begins with a backslash (\),

BIM-EDIT treats STR as an Extended Search Pattern, with certain characters having special meanings:

- ? matches any single character
- * matches any sequence of zero or more characters
- @ matches zero or more instances of the character that follows the @
- ! matches the beginning of the zone
- < matches the beginning of the zone or any non-alphanumeric character
- > matches the end of the zone or any non-alphanumeric character
- | separates two patterns, EITHER of which may match for a successful search
- + separates two patterns, BOTH of which must match for a successful search
- ~ (NOT) reverses line-by-line success/failure of the pattern following it
- % marks the position on the line where the match is considered to occur
- \ suppresses the special meaning of the character that follows the \

See Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual for more information about Extended Search Patterns.

If STR is not specified on the command, BIM-EDIT uses as default operand values those from previous search commands. Using this feature, you can easily request searches which are similar or identical to the last one. For example, after a LOCATEU command which failed because the instance is forward from the current line, you need only enter

`=> locate`

to repeat exactly the same command in the forward direction. To achieve this, BIM-EDIT stores a previous value for the STR, COL, CASE, and ZONE operands of the search commands (EXAMINE, FIND, FINDUP, LOCATE, LOCATEU, NFIND, NFINDUP, QUALIFY, or SCAN). These previous values will be inherited by operands omitted from a search command which does not specify STR. Whenever a search command specifies STR, the previous values are set to those specified on that command or "not specified".

Use in a Procedure

Return Codes:

- | | |
|----|----------------------------------|
| OK | Successful. |
| AT | Command interrupted by ATTN key. |
| NF | String not found. |

LOCATEU sets the predefined variable SSDCOL1 to the column number where the match occurred (if in split-screen mode, and if the lower logical screen is primary, SSDCOL2 is set instead).

Example

Screen before LOCATEU command

```

=> locateu 4096
LIST 4217.IOROUT                                SESS=A 1( 1) LINE= 38( 256)
-----1-----2-----3-----4-----5-----6-----7--
*====*      STH      R1,IOLNGTH      SAVE IN I/O CONTROL FIELD
*====*      L        R1,PM2          GET EXTENT INFO PARM
*====*      MVC      IOEXTENT,0(R1)   USE FILE EXTENT 1 INFO FOR RELOCATE
*====*      BAL      R11,RELOCATE     RELOCATE CHANNEL PROGRAM
*====*      BAL      R11,CALCSEEK     CALCULATE SEEK ADDRESS
*====*      BAL      R11,EXCPDISK     DO I/O
*====*      B        RETURN          RETURN TO CALLER
*====* *****
*====*      SUBROUTINES
*====*      1) RMZPQAR - HANDLES INPUT FROM CARDS
*====*      2) RMZPQXR - HANDLES INPUT FROM TAPE
*====*      3) RMZPQXA - HANDLES INPUT FROM SCREEN
*====*      4) RMZPQXM - HANDLES INPUT FROM MEMBER
*====*      5) RMZPRS1 - PARSES INPUT STRING, TYPE 1
*====*      6) RMZPRS2 - PARSES INPUT STRING, TYPE 2
*====*      7) RMZPRS3 - PARSES INPUT STRING, TYPE 3
*====*
*====*      ALL SUBROUTINES RECEIVED ADDRESSABILITY TO
*====* *****

```

Screen after LOCATEU command

```

=>
LIST 4217.IOROUT                                SESS=A 1( 1) LINE= 34( 256)
-----1-----2-----3-----4-----5-----6-----7--
*====*      MVC      TXLBLKSZ,=H'4096' SET CONSTANT TXLFL RECORD SIZE
*====*      L        R1,PM4          GET BLOCK COUNT ADDRESS
*====*      LH       R1,0(R1)        GET BLOCK COUNT
*====*      MH       R1,TXLBLKSZ     COMPUTE NUMBER OF BYTES NEEDED
*====*      STH      R1,IOLNGTH      SAVE IN I/O CONTROL FIELD
*====*      L        R1,PM2          GET EXTENT INFO PARM
*====*      MVC      IOEXTENT,0(R1)   USE FILE EXTENT 1 INFO FOR RELOCATE
*====*      BAL      R11,RELOCATE     RELOCATE CHANNEL PROGRAM
*====*      BAL      R11,CALCSEEK     CALCULATE SEEK ADDRESS
*====*      BAL      R11,EXCPDISK     DO I/O
*====*      B        RETURN          RETURN TO CALLER
*====* *****
*====*      SUBROUTINES
*====*      1) RMZPQAR - HANDLES INPUT FROM CARDS
*====*      2) RMZPQXR - HANDLES INPUT FROM TAPE
*====*      3) RMZPQXA - HANDLES INPUT FROM SCREEN
*====*      4) RMZPQXM - HANDLES INPUT FROM MEMBER
*====*      5) RMZPRS1 - PARSES INPUT STRING, TYPE 1
*====*      6) RMZPRS2 - PARSES INPUT STRING, TYPE 2

```

LOGI

Use LOGI to write a specified line (sometimes referred to as the "immediate" line) to \$LOG.

Required Operands	
LINE	is the line to write to the log.

Use in a Procedure

LOGI normally returns OK. (See SIBRETC D.)

Example

Write a line to the log:

```
=> logi '-- about to clean up library --'
```

LOGOFF

Use LOGOFF when, for the time being, you are done with BIM-EDIT.

LOGOFF may also be entered as LOGOF.

LOGOFF has no operands.

A date and time message will be written to \$LOG.

The entire environment is retained so that at the next logon all the active sessions will be available. In other words, it is not necessary to end sessions before logging off.

Use in a Procedure

Return codes are not applicable, since BIM-EDIT terminates.

Batch utility jobs can also logoff by reaching the end of file line "/*" or by processing the EXIT command.

Example

Log off:

```
=> logoff
```

LOWERCAS

Use LOWERCAS to translate current session text from uppercase letters to lowercase letters within a specified column range for a specified number of lines starting with the current line.

LOWERCAS may also be entered as LOWER or LOW.

Optional Operands	
FCT	specifies the number of lines for which text is to be translated. If FCT is specified as an asterisk (*), text will be translated for all lines through the end of the session. If FCT is not specified, text will be translated for the current line only.
ZONE	is the column range in which to translate text. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to translate text within columns 11 to 66, specify ZONE as "11-66". ZONE=5 is the same as ZONE=5-5, ZONE=5-* is the same as ZONE=5-253 and ZONE=-5 is the same as ZONE=1-5. If ZONE is not specified, text is translated in the current session zone (see the ZONE operand of the DEFINE, ALTER, and SESS commands.)

The UPPERCAS command can be used to translate from lowercase to uppercase.

LOWERCAS is the command line equivalent of the LCA W command.

Use in a Procedure

LOWERCAS normally returns OK. (See SIBRETC D.)

Examples

Translate to lowercase through the end of the member:

```
=> lowercas *
```

Translate 10 lines to lowercase:

```
=> low 10
```

MAIL

Use MAIL to send the text of a member as a message to another user. (MAILI is often used for single-line messages.)

Required Operands	
DEST	is the user to send the message to. (The LIBRARYU command will produce a listing of all users.)

Optional Operands	
MEM	is the member containing the message to be sent. If MEM is not specified, the "last referenced member" is used. MEM can also be "\$STACK" to send the contents of the stack area or "/" to send the current session.
TITLE	is a brief comment about or description of the message. TITLE can be up to 40 characters in length. If TITLE is not specified, the first 40 characters of the first line of text is used. The message title is shown on the display created by the LIBRARYQ command.
NOPUR	indicates whether or not this message will appear in the users outgoing mail queue. If it appears, the message may be deleted from the outgoing mail queue before it is opened by the DEST user. The parameter may be specified as "YES" or "NO". "YES" means that the message will not appear in the outgoing queue. "NO" is the default and the message will appear in the outgoing mail queue.
ACK	indicates whether or not this message, when opened, will cause a response (message) to be returned to the sending user. The response identifies the DEST, message id, time and date sent, the 40 characters of TITLE (see TITLE above), and the fact that it "has been opened". The parameter may be specified as "YES" or "NO". "YES" means that a response is REQUIRED. "NO" is the default.
NTFY	indicates that this mail is a "notify" type message that can be displayed on the line 2 message area of the DEST users screen if they have the AUTOMAIL user option defined. Refer to the DEFINEU command for more information on the AUTOMAIL option. This parameter can be specified as "YES", "PURGE" or "NO". "PURGE" means that the message is automatically purged from the DEST users set-aside queue after being viewed via the AUTOMAIL user option. "NO" is the default.

MAIL queues a message for the DEST user. The message text is copied from the MEM member. The MEM member is immediately available for reuse or deletion. The DEST

user is informed of the queued message by a highlighted indicator in his information line. The DEST user can then issue the OPEN command to view the message.

The message is assigned a unique five digit number. This number is shown as part of the information line when the message is displayed. It is also shown on the LIBRARYQ display.

LIBRARYQ will list all messages sent by you that have not been displayed by the DEST user. These messages are listed in the 'OUTGOING MAIL' section of the display. You can display any such message by issuing the OPEN command followed by the message number or by using the LCA L command. The message can be purged by issuing the DISCARD command while viewing the message. You can also purge such a message by issuing the PURGEQ command followed by the message number or by using the LCA P command. If a message is purged before the DEST user opens the message, the DEST user receives no indication that a message was ever sent.

MAIL logs the message text to \$MAIL. \$MAIL can be displayed at any time by entering:

```
=> list $mail
```

MAIL sets the "last referenced member".

When Valid

The user must have LIST access level for the MEM library.

Use in a Procedure

Return Codes:

- | | |
|----|---|
| OK | Successful. |
| NF | One of the following: |
| | <ul style="list-style-type: none">• Member not found.• User not found. |
| SC | Inadequate access level. |

MAIL sets the TXM variables to the attributes of MEM.

Examples

Send the message in member MEMO to user SJK:

```
=> mail sjk,memo
```

Send the message in member NOTE with a title to user GOP:

```
=> mail gop,note,'trip itinerary'
```

MAILI

Use MAILI to send a specified line (sometimes called the "immediate" line) as a message to another user.

Required Operands	
DEST	is the user to send the message to. (The LIBRARYU command will produce a listing of all users.)
TEXT	is the message text to send to the DEST user. TEXT can be up to 250 characters in length.

Optional Operands	
NOPUR	indicates whether or not this message will appear in the users outgoing mail queue. If it appears, the message may be deleted from the outgoing mail queue before it is opened by the DEST user. The parameter may be specified as "YES" or "NO". "YES" means that the message will not appear in the outgoing queue. "NO" is the default and the message will appear in the outgoing mail queue.
ACK	indicates whether or not this message, when opened, will cause a response (message) to be returned to the sending user. The response identifies the DEST, message id, time and date sent, the 40 characters of TITLE, and the fact that it "has been opened". The parameter may be specified as "YES" or "NO". "YES" means that a response is REQUIRED. "NO" is the default.
NTFY	indicates that this mail is a "notify" type message that can be displayed on the line 2 message area of the DEST users screen if they have the AUTOMAIL user option defined. Refer to the DEFINEU command for more information on the AUTOMAIL option. This parameter can be specified as "YES", "PURGE" or "NO". "PURGE" means that the message is automatically purged from the DEST users set-aside queue after being viewed via the AUTOMAIL user option. "NO" is the default.

The message title displayed by LIBRARYQ is set to the first 40 characters of TEXT.

MAILI is used for sending a short message without the inconvenience of creating and editing a member. After the message has been sent, it behaves exactly as if a single-line member had been sent using MAIL.

See the MAIL command for more discussion of mail features.

Use in a Procedure

MAILI normally returns OK. (See SIBRETCO.)

Example

Send a single-line message:

```
=> maili mb, 'Thanks.  See you at the meeting.'
```

MAILSESS

Use MAILSESS to send the current session as a message to another user.

Required Operands	
DEST	is the user to send the message to. (The LIBRARYU command will produce a listing of all users.)

MAILSESS queues a message for the DEST user. The message text is the current session. The DEST user is informed of the queued message by a highlighted indicator in his information line. The DEST user can then issue the OPEN command to view the message.

The message is assigned a unique five digit number. This number is shown as part of the information line when the message is displayed. It is also shown on the LIBRARYQ display.

LIBRARYQ will list all messages sent by you that have not been displayed by the DEST user. These messages are listed in the 'OUTGOING MAIL' section of the display. You can display any such message by issuing the OPEN command followed by the message number or by using the LCA L command. The message can be purged by issuing the DISCARD command while viewing the message. You can also purge such a message by issuing the PURGEQ command followed by the message number or by using the LCA P command. If a message is purged before the DEST user opens the message, the DEST user receives no indication that a message was ever sent.

MAILSESS logs the message text to \$MAIL. \$MAIL can be displayed at any time by entering:

```
=> list $mail
```

Use in a Procedure

Return Codes:

OK	Successful.
NF	User not found.

Example

Send the current session to user SJK:

```
=> mailsess sjk
```

MERGE

Use MERGE to replace blank characters in current session text with characters from \$STACK for a specified number of lines starting with the current line.

Optional Operands	
FCT	specifies the number of lines to update. If FCT is specified as an asterisk (*), all of the lines through the end of the session will be updated. If FCT is not specified, FCT is set to the number of lines in \$STACK. IF FCT is greater than the number of lines in \$STACK, MERGE will process \$STACK as many times as are necessary to exhaust the number of lines specified by FCT. This feature is often used to merge a single \$STACK line against a number of session lines.

MERGE updates session lines starting with the current line by overlaying non-blank text from \$STACK lines onto the corresponding session lines. The overlay occurs for a given line position only if the position is blank in the session line.

MERGE assembles text lines from more than one source. It can be used to make side-by-side (newspaper-style) columns or to apply a static pattern of characters over a varying one.

MERGE is the command line equivalent of the LCA J command. The LCA G command also performs the MERGE function.

Use in a Procedure

MERGE normally returns OK. (See SIBRETC D.)

Example

\$STACK before (and after) MERGE

```
=>
LIST $STACK
-----1-----2-----3-----4-----5-----6-----7-----
-- TOP OF $STACK --

                                For, like strains of martial music,
                                Their mighty thoughts suggest
                                Life's endless toil and endeavor;
                                And tonight I long for rest.

-- END OF $STACK --
```

EDIT session before MERGE

```
=> merge
EDIT  5624.DAYISDON                      SESS=A 1( 2) LINE=    0(   10)
----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----|-----
-- TOP OF MEMBER --
Not from the grand old masters,
Not from the bards sublime,
Whose distant footsteps echo
Through the corridors of Time.

Read from some humbler poet,
Whose songs gushed from his heart,
As showers from the clouds of summer,
```

EDIT session after MERGE

```
=>
EDIT  5624.DAYISDON                      SESS=A 1( 2) LINE=    0(   10)
----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----|-----
-- TOP OF MEMBER --
Not from the grand old masters,          For, like strains of martial music,
Not from the bards sublime,              Their mighty thoughts suggest
Whose distant footsteps echo             Life's endless toil and endeavor;
Through the corridors of Time.            And tonight I long for rest.

Read from some humbler poet,
Whose songs gushed from his heart,
As showers from the clouds of summer,
```

MESSAGE (VSE only)

Use MESSAGE to create a display session of the explanation of one or more VSE Messages contained in the VSE.MESSAGES.ONLINE Dataset (IESMSGs).

MESSAGE may also be entered as MSG.

Required Operands	
MSG	<p>is the message id to be displayed. If the last position of MSG is an asterisk (*), a list is generated of all messages that match the characters prior to the (*). Example: msg dfh* will list all messages that begin with 'dfh'.</p> <p>You can also display the text of a message by entering the MESSAGE command without specifying the MSG operand and placing the cursor on a message id contained in the text currently being displayed.</p>
CTL	<p>used to open or close the IESMSGs dataset. The use of this operand requires that the user have ADM command security. CTL can be entered as OPEN or CLOSE. If the CTL operand is specified, the MSG operand will be ignored.</p>

If a list of messages is generated as a result of using the (*) on the MSG operand, you can display the text of a specific message by using the LCA L command.

If the (*) is not specified on the MSG operand, the display is generated without an LCA.

There are several special messages provided by IBM that may be useful:

VSAMCLOS	Contains a description of the VSAM CLOSE return codes and error codes that can appear in various VSAM messages.
VSAMOPEN	Contains a description of the VSAM OPEN return codes and error codes that can appear in various VSAM messages.
VSAMREQU	Contains a description of the VSAM request macro return codes and error codes.
VSAMXXCB	Contains a description of the VSAM return codes and error codes that can be returned from GENCB, SHOWCB, and TESTCB macros.

When Valid

Your system must be on VSE/ESA 2.1 or above.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Requested message id not found.
IV	Error returned from VSAM interface.

Examples

Create a display session for the description of message 1C39I.

```
=> msg 1c39i
```

Resulting Display of Message 1C39I

```
=>
DISP  -> msg 1c39i                      SESS=A 3( 3) LINE=    0(   11)
-----1-----2-----3-----4-----5-----6-----7-----
-- TOP OF DISPLAY --
1C39I      COMMAND PASSED TO subsystem

          Explanation: The last attention command is passed to VSE/ICCF,
          VTAM, or VSE/POWER, and the attention routine is ready to handle
          the next attention command.

          System Action:  None.

          Programmer Response:  None.

          Operator Response:  None.
-- END OF DISPLAY --
```

Create a display session for the description of all messages that start with the characters VSAM.

```
=> msg vsam*
```

Resulting Display of Message 1C39I

```
=>
DISP  -> msg vsam*                      SESS=A 4( 4) LINE=    0(   5)
-----1-----2-----3-----4-----5-----6-----7-----
*==== -- TOP OF DISPLAY --
*==== VSAMCLOS  - Error Codes from CLOSE or TCLOSE
*==== VSAMOPEN  - Error Codes from OPEN
*==== VSAMREQU  - Error Codes from Request Macros
*==== VSAMRESN  - IDCAMS Return and Reason Codes
*==== VSAMXXCB  - Error Codes from GENCB/MODCB/SHOWCB/TESTCB
-- END OF DISPLAY --
```

MOVE

Use MOVE to change the name of a member. MOVE is also used to move a member from one library to another.

The MOVE command performs the same function as the RENAME command.

Required Operands	
SRC	is the existing member name.
DEST	is the new member name. A member having the destination name must not already exist.
PSWD	If the member is password protected, the password must be entered. The password may be up to 8 characters in length consisting of any characters. The moved member remains password protected.

If an asterisk (*) is entered as part of DEST, the asterisk is replaced with the SRC member name. This offers an easy way of moving a member from one library to another or adding a suffix or prefix to a name without keying the member name twice (see Examples).

MOVE sets the "last referenced member" to DEST.

When Valid

The user must have DEF access level for both the SRC library and the DEST library. SRC cannot be currently undergoing editing or listing. SRC cannot be either the slave or the master in a checkout relationship.

Use in a Procedure

Return Codes:

- | | |
|----|---|
| OK | Successful. |
| CK | SRC member is part of a checkout relationship. |
| DP | DEST member already exists. |
| ED | SRC member is being edited. |
| LI | SRC member has active LIST session(s). |
| NF | One of the following: <ul style="list-style-type: none">• SRC member not found.• DEST library not found. |

MOVE sets the TXM variables to the attributes of DEST.

MOVE is implemented as the system procedure BIPMOVE.

Examples

Rename SYRINT1 to TKRINT1:

```
=> move syrint1,tkrint1
```

Move member XMRL21 to library 4216:

```
=> move xmrl21,4216.xmrl21      -or-  
=> move xmrl21,4216.*
```


NEXT

Use NEXT to position the current session forward (i.e toward the bottom) a specified number of lines.

NEXT may also be entered as N, DOWN, or DO.

Optional Operands	
FCT	specifies the number of lines to move the position by. If FCT is not specified, BIM-EDIT will position forward one line.

It is not an error if FCT exceeds the number of lines to the bottom of the session.

Use in a Procedure

Return Codes:

OK	Successful
EF	Already positioned at the last line of the session (this only occurs when NEXT is executed from a procedure).

Examples

Position forward 29 lines:

```
=> next 29
```

Position forward 1 lines:

```
=> n
```

NFIND

Use NFIND to position the current session forward from the current line to the first line which does not have a specified string of characters at a specified column. That is, NFIND looks for the absence of a string of characters; it succeeds as soon as it doesn't find the string.

NFIND may also be entered as NF.

Optional Operands	
STR	is the string whose absence is to be searched for. Up to 72 characters may be specified. If the value of STR begins with the backslash character (\), the search will use Extended Search Pattern rules, as described below. If STR is not specified, the value for STR and any other operands not specified will be inherited from previous search commands, also as described below.
COL	specifies the column to search at. COL must be a value from 1 to 253. If COL and STR are not specified, the COL specification is inherited from previous search commands. If COL, after any inheriting, is not specified, the current session FCOL value is used (see the DEFINE, ALTER, and SESS commands).
CASE	specifies whether upper/lower case should be considered when determining matches. If specified as "U", case is ignored ("the" matches "The"). If specified as "M", both case and letters must match. If CASE and STR are not specified, the CASE specification is inherited from previous search commands, as described below. If CASE, after any inheriting, is not specified, "U" is assumed.

The search begins with the line following the current line and proceeds until a line is found that does not contain the specified match at the specified column. An error message will display if such a line is not found.

NFIND is used to find the next line at which a consecutive series of values changes. It is used with text lines that contain very consistent data. For example, after text lines have been sorted, NFIND can be used to find the next "break" in the key columns.

After NFIND, the CURSOR command will place the cursor at the search column.

If your terminal is running under control of SNA, you may interrupt a long NFIND command by pressing the ATTN key.

If the value of STR begins with a backslash (\), BIM-EDIT treats STR as an Extended Search Pattern, with certain characters having special meanings:

- ? matches any single character
- * matches any sequence of zero or more characters
- @ matches zero or more instances of the character that follows the @
- ! matches before the COL column

- < matches before the COL column or any non-alphanumeric character
- > matches the end of the line or any non-alphanumeric character
- | separates two patterns, EITHER of which may match for a successful search
- + separates two patterns, BOTH of which must match for a successful search
- ~ (NOT) reverses line-by-line success/failure of the pattern following it
- % marks the position on the line where the match is considered to occur
- \ suppresses the special meaning of the character that follows the \

See Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual for more information about Extended Search Patterns.

If STR is not specified on the command, BIM-EDIT uses as default operand values those from previous search commands. Using this feature, you can easily request searches which are similar or identical to the last one. For example, after an NFIND command which failed because the absence is backward from the current line, you need only enter

```
=> nfindup
```

to repeat exactly the same command in the backward direction. To achieve this, BIM-EDIT stores a previous value for the STR, COL, CASE, and ZONE operands of the search commands (EXAMINE, FIND, FINDUP, LOCATE, LOCATEU, NFIND, NFINDUP, QUALIFY, or SCAN). These previous values will be inherited by operands omitted from a search command which does not specify STR. Whenever a search command specifies STR, the previous values are set to those specified on that command or "not specified".

Use in a Procedure

Return Codes:

- OK Successful.
- AT Command interrupted by ATTN key.
- NF Line without character string not found.

NFIND sets the predefined variable SSDCOL1 to COL if the search is successful (if in split-screen mode, and if the lower logical screen is primary, SSDCOL2 is set instead).

Examples

Search for the first line that does not start with an asterisk (assume that the session FCOL value is set at 1):

```
=> nfind *
```

Search for the first line that does not contain either the character string "DS" or the character string "DC" starting in column 10:

```
=> nf \DS|DC,10
```

Search for the absence of the same string as the last search:

```
=> nf
```

NFINDUP

Use NFINDUP to position the current session backward from the current line to the first line which does not have a specified string of characters at a specified column. That is, NFINDUP looks for the absence of a string of characters; it succeeds as soon as it doesn't find the string.

NFINDUP may also be entered as NFINDU, NFUP, or NFU.

Optional Operands	
STR	is the string whose absence is to be searched for. Up to 72 characters may be specified. If the value of STR begins with the backslash character (\), the search will use Extended Search Pattern rules, as described below. If STR is not specified, the value for STR and any other operands not specified will be inherited from previous search commands, also as described below.
COL	specifies the column to search at. COL must be a value from 1 to 253. If COL and STR are not specified, the COL specification is inherited from previous search commands. If COL, after any inheriting, is not specified, the current session FCOL value is used (see the DEFINE, ALTER, and SESS commands).
CASE	specifies whether upper/lower case should be considered when determining matches. If specified as "U", case is ignored ("the" matches "The"). If specified as "M", both case and letters must match. If CASE and STR are not specified, the CASE specification is inherited from previous search commands, as described below. If CASE, after any inheriting, is not specified, "U" is assumed.

The search begins with the line preceding the current line and proceeds upward until a line is found that does not contain the specified match at the specified column. An error message will display if such a line is not found. (Since it is not possible to position AFTER the last line in a session, any occurrence of STR on the last line will be ignored by NFINDUP.)

After NFINDUP, the CURSOR command will place the cursor at the search column.

If your terminal is running under control of SNA, you may interrupt a long NFINDUP command by pressing the ATTN key.

If the value of STR begins with a backslash (\), BIM-EDIT treats STR as an Extended Search Pattern, with certain characters having special meanings:

- ? matches any single character
- * matches any sequence of zero or more characters
- @ matches zero or more instances of the character that follows the @
- ! matches before the COL column
- < matches before the COL column or any non-alphanumeric character

- > matches the end of the line or any non-alphanumeric character
- | separates two patterns, EITHER of which may match for a successful search
- + separates two patterns, BOTH of which must match for a successful search
- ~ (NOT) reverses line-by-line success/failure of the pattern following it
- % marks the position on the line where the match is considered to occur
- \ suppresses the special meaning of the character that follows the \

See Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual for more information about Extended Search Patterns.

If STR is not specified on the command, BIM-EDIT uses as default operand values those from previous search commands. Using this feature, you can easily request searches which are similar or identical to the last one. For example, after an NFINDUP command which failed because the absence is backward from the current line, you need only enter

```
=> nfind
```

to repeat exactly the same command in the forward direction. To achieve this, BIM-EDIT stores a previous value for the STR, COL, CASE, and ZONE operands of the search commands (EXAMINE, FIND, FINDUP, LOCATE, LOCATEU, NFIN, NFINDUP, QUALIFY, or SCAN). These previous values will be inherited by operands omitted from a search command which does not specify STR. Whenever a search command specifies STR, the previous values are set to those specified on that command or "not specified".

Use in a Procedure

Return Codes:

- OK Successful.
- AT Command interrupted by ATTN key.
- NF Line without character string not found.

NFINDUP sets the predefined variable SSDCOL1 to COL if the search is successful (if in split-screen mode, and if the lower logical screen is primary, SSDCOL2 is set instead).

Examples

Search backwards for the first line that does not start with an asterisk (assume that the session FCOL value is set at 1):

```
=> nfindup *
```

Search backwards for the first line that does not contain either the character string "DS" or the character string "DC" starting in column 10:

```
=> nfu \DS|DC,10
```

Search backwards for the absence of the same string as the last search:

```
=> nfu
```

OPEN

Use OPEN to create a LIST session of a mail message.

OPEN may also be entered as OP.

Optional Operands	
ID	is a five digit number assigned to the message. If ID is not specified, the first message in the incoming-mail queue will be displayed.

OPEN can also be invoked by entering the L command in the LCA area of a LIBRARYQ display.

BIM-EDIT maintains three mail queues:

Incoming-Mail: Messages you have received but not displayed

Mail-set-aside: Messages you have displayed but not purged

Outgoing-Mail: Messages you have sent but have not been displayed by the DEST user

The LIBRARYQ command lists entries for all three queues.

If the incoming-mail queue contains any messages, the word "MAIL" will appear on the information line (if a session exists) or on the home screen body.

When a message is OPENed from the incoming-mail queue, it is transferred to the mail-set-aside queue while the message display session is being created. If the session is then terminated with the END command, the message can be displayed again using OPEN with the ID. If the session is terminated with the DISCARD command, the message will be deleted from the mail-set-aside queue.

When you OPEN a message from your incoming-mail queue, the message text is logged to your \$MAIL. \$MAIL can be displayed at any time by entering:

```
=> list $mail
```

Use in a Procedure

Return Codes:

OK	Successful.
NF	Message not found.

Examples

Display the first message in the incoming-mail queue:

```
=> open
```

Display message #2604:

```
=> open 2604
```

OVERLAY

Use OVERLAY to replace current session text with a specified string of characters at a specified column for a specified number of lines starting at the current line.

OVERLAY may also be entered as OVLY or O.

Required Operands	
COL	is the column number where the literal is to be overlaid. COL may range from 1 through 253.
LIT	is the literal to be overlaid. LIT may be up to 72 characters in length

Optional Operands	
FCT	specifies the number of lines to overlay. If FCT is specified as an asterisk (*), all of the lines through the end of the session will be overlaid. If FCT is not specified, only the current line will be overlaid.

Use in a Procedure

OVERLAY normally returns OK. (See SIBRETC D.)

Example

EDIT session before OVERLAY command

```
=> overlay 4,s20 ,9
EDIT  DOC.C                      SESS=A 1( 1) LINE=      8(    93)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*      AC N  = PMA AC;
*====*      NAME  = PMA NAME;
*====*      ADDR1 = PMA ADDR1;
*====*      ADDR2 = PMA ADDR2;
*====*      CITY  = PMA CITY;
```

EDIT session after OVERLAY command

```
=>
EDIT  DOC.C                      SESS=A 1( 1) LINE=      8(    93)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*      S20 AC N  = PMA AC
*====*      S20 NAME  = PMA NAME
*====*      S20 ADDR1 = PMA ADDR1;
*====*      S20 ADDR2 = PMA ADDR2;
*====*      S20 CITY  = PMA CITY;
```

PASSWORD

Use PASSWORD to alter your permanent user password.

PASSWORD may also be entered as PASS, PAS, or PSWD.

Required Operands	
PSWD	is the new password. PSWD can be up to 8 characters in length.

Subsequent logons will require the new password.

IF the BIM-EDIT PASSWORD Verification Feature has been activated, this command will generate a MAPF screen requesting that you enter your current BIM-EDIT password before the new password will take affect. Refer to Chapter 7 of the BIM-EDIT System Reference Manual for information on how to activate this feature.

When the PASSWORD Verification Feature is active, the following MAPF screen is generated:

<div><div>PASSWORD</div><div>Change BIM-EDIT PASSWORD</div><div>-----</div><div>Enter new password:</div><div>and old password:</div></div>

Use in a Procedure

PASSWORD normally returns OK. (See SIBRETCO.) If the PASSWORD Verification Feature has been activated, you must have ADM security to use the PASSWORD command in a procedure.

Example

Change your password to X593:

```
=> password x593
```


POSITION

Use POSITION to position the current session to a specified line number.

POSITION may also be entered as POS or P.

Required Operands	
LINE	is the line number. If the line number is omitted BIM-EDIT will attempt to position the current screen based on the position of the cursor.

The information line LINE= field shows the line number and, in parentheses, the total lines in the session. For example, "LINE= 12(100)" denotes that the current line is the 12th line of 100 lines.

POSITION is the command line equivalent of the LCA / command.

Use in a Procedure

Return Codes:

- OK Successful.
- EF Line number exceeds number of lines in session.

Example

Position at line number 245:

```
=> position 245
```

LIST session after POSITION command

```
=>
LIST ED51A.BIZUSRF                      SESS=A 2( 2) LINE= 245( 13753)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----
*====*  (( ... Left justify these text lines .....
*====*  ),) ... Right justify these text lines .....
*====*
*====* Chapter 6. Advanced Techniques .....
*====* Multiple Operations on a Screen .....
*====* Sequences of Commands .....
*====* Split-Screen Operation .....
*====* Logical Tabbing .....
*====* Hexadecimal Display and Editing .....
```

PRINT

Use PRINT to print the text of a member.

PRINT can also be entered as RELIST.

Optional Operands	
MEM	is an existing member. If MEM is not specified, the "last referenced member" is used. For online print on VSE, MEM can be specified as "/" to print the current session, "\$STACK" to print your STACK area, "\$MAIL" to print your MAIL queue, or "\$LOG" to print your LOG area.
CLASS	is the job class for the resulting print. If CLASS is not specified, "A" is used.
FNO	is the form number for the resulting print. If FNO is not specified, the operating system's default form number will be used. (For VSE, the default is blanks.)
CASE	If CASE is specified as "U", text will be translated to upper case. If CASE is specified as "M", text will not be translated. If CASE is not specified, "M" is assumed.
FMT	specifies the format to use when printing lines. Specify "NO" if exact line images are to be printed. Specify "YES" if lines are to be prefixed with line numbers. If FMT is not specified, a value of "YES" is assumed.
PGSIZ	printed page size. A value specified for PGSIZ will determine the number of lines to be printed on a page. If PGSIZ is not specified the number of lines per page will be the value of MMPPGSIZ.
PSWD	If the member is password protected, the password must be entered before the print request is allowed. The password may be up to 8 characters in length consisting of any character.

Optional Operands (VSE only)	
RJEID	RJEID may be specified as a value of "1" to "250". RJEID may also be specified as "R001" to "R250" but in this case must be 4 characters long.
JSEP	If JSEP is specified as "0", no page separator pages will be printed. A value of "1" to "9" will cause POWER to print as many page separators as the value specified.
COPY	COPY may be specified as a value from 1 to 255 to indicate the number of copies to be printed. The default is 1.

Optional Operands (VSE only, continued)	
DESTNODE	Is a 1 to 8 character name of a nodeid to which POWER is to route the listing. If omitted the destination will be the system on which BIM-EDIT is running.
DESTUSER	Is a 1 to 8 character name of a destination userid or "ANY". If omitted DESTUSER will default to "LOCAL". DESTUSER may also be used to specify a RJEID value but only if the "Rnnn" format is used. DESTUSER will always override any RJEID specified.
UINF	is the 16 character user information to be associated with the POWER LST output.

On VSE, PRINT can be used both under online BIM-EDIT and in batch utility. Direct online print (and printing the current session) is only available for VSE/SP release 2.1 and later. For prior releases, if PRINT is invoked online, it submits a job that invokes the PRINT procedure in batch utility.

On MVS, PRINT submits a job that invokes the PRINT procedure in batch utility.

The PRINT command, both online and batch, determines the number of lines to print on a page based on the value of the variable MMPPGSIZ.

PRINT sets the "last referenced member".

Use in a Procedure

Return Codes:

OK	Successful.
NF	Member not found.
SC	Inadequate access level.

PRINT sets the TXM variables to the attributes of MEM.

PRINT is implemented as the system procedure BIPPRNT. PRINT handles the actual printing by invoking the PRINTF command.

Example

From online, print the member XIL0250. Translate to upper case:

```
=> print xil0250,case=u
```

From batch, print the members RMGL300 and RMGL400:

VSE Job Control

```
// JOB PRINT
// EXEC BIMUTIL
LOGON $SYS,$SYS
PRINT RMGL300
PRINT RMGL400
/*
/ &
```

MVS Job Control

```
//PRINTJOB JOB    ....  
//BIMUTIL  EXEC  PGM=BIMUTIL  
//STEPLIB  DD    DSN=BIMEDIT.LOADLIB,DISP=SHR  
//SYSPRINT DD    SYSOUT=A  
//SYSIN    DD    *  
LOGON $SYS,$SYS  
PRINT RMGL300  
PRINT RMGL400  
/*
```

PROCESS

Use PROCESS to SUBMIT, COMPILE, or EXECUTE a member based upon its type.

PROCESS may also be entered as PROC.

Optional Operands	
MEM	is the member to be processed. If MEM is not specified, the "last referenced member" is used.
PSWD	If the member is password protected, the password must be entered before the request is allowed. The password may be up to 8 characters in length consisting of any characters.

PROCESS can also be invoked by entering the S command in the LCA area of a LIBRARY display.

PROCESS is often customized to support site-specific member types. Check with your System Administrator. The following table shows what PROCESS will do for each member type as BIM-EDIT is distributed:

Type	Command	Type	Command
ASM	COMPILE	PLI	COMPILE
COBOL	COMPILE	PROC	EXECUTE
FORT	COMPILE	RPG	COMPILE
JCL	SUBMIT		

Since PROCESS will make use of the "last referenced member", the following string is a good candidate for a PF key assignment:

```
=> save;process
```

PROCESS sets the "last referenced member".

When Valid

The user must have LIST access level for the MEM library.

Use in a Procedure

Return Codes:

OK	Successful.
ED	Member is being edited.
NF	Member not found.
SC	Inadequate access level.

PROCESS sets the TXM variables to the attributes of MEM.

PROCESS is implemented by the system procedure BIPPROC.

Example

Process the member OMJBKPG:

```
=> process omjbkpg
```

PROPAGAT

Use PROPAGAT to copy current session text characters from one column range to another for a specified number of lines starting at the current line.

PROPAGAT may also be entered as PROP.

Required Operands	
SCOL	is the column range in which the source text resides. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to copy text existing in columns 11 through 20, specify SCOL as "11-20". SCOL=5 is the same as SCOL=5-5, SCOL=5-* is the same as SCOL=5-253 and SCOL=-5 is the same as SCOL=1-5.
DCOL	is the destination column number. DCOL may range from 1 through 253. DCOL may be within the column range specified by SCOL.
Optional Operands	
FCT	specifies the number of lines to update. If FCT is specified as asterisk (*), all lines through end of the session are updated. If FCT is not specified, only the current line is updated.

The text residing in the SCOL column range will be copied to the DCOL location.

To delete session text at a column, use SHIFT.

Use in a Procedure

PROPAGAT normally returns OK. (See SIBRETC D.)

Example

EDIT session before PROPAGAT command

```
=> propagat 17-24,44,*
EDIT  EDTP.TEMP                                SESS=A 1( 1) LINE=    0( 72)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF MEMBER --
*====* COPY $SYS.HELP.ADD                        EDTP.HP-
*====* COPY $SYS.HELP.ALT                        EDTP.HP-
*====* COPY $SYS.HELP.ALTER                      EDTP.HP-
```

EDIT session after PROPAGAT command

```
=>
EDIT  EDTP.TEMP                                SESS=A 1( 1) LINE=    0( 72)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF MEMBER --
*====* COPY $SYS.HELP.ADD                        EDTP.HP-ADD
*====* COPY $SYS.HELP.ALT                        EDTP.HP-ALT
*====* COPY $SYS.HELP.ALTER                      EDTP.HP-ALTER
```

PURGE

Use PURGE to delete a member.

PURGE may also be entered as PUR.

Required Operands	
MEM	is the member to be purged. If MEM is specified as an asterisk (*), the "last referenced member" is purged.

Optional Operands	
OPT	may be specified as NOPGCTL to disable the purge control option if MMPPGCTL is set to 1. If the ARCHIVE feature is installed it may be specified as FORCE to allow purging of members contained in either the \$SIT.GEN or \$SIT.ARCHIVE libraries. These libraries are normally maintained using the ARCHIVE and RECOVER commands. You must have ADM level security to use the FORCE and NOPGCTL options.
PSWD	If the member is password protected, the password must be entered. The password may be up to 8 characters in length consisting of any characters.

PURGE can also be invoked by entering the P command in the LCA area of a LIBRARY display.

PURGE sets the "last referenced member". This is typically not useful because no further processing can be done to the member.

If your System Administrator has set predefined variable MMPPGCTL to "1", members can be retrieved for a short period of time after an erroneous PURGE. Otherwise, the only possible way to retrieve a member after a PURGE is to use RESTORE from a backup tape.

When the ARCHIVE feature is installed, any existing generations in the \$SIT.GEN library are moved to the \$SIT.ARCHIVE library as well as the source for MEM. In this way the integrity of the ARCHIVE facility is maintained.

When Valid

The user must have DEF access level for the MEM library. MEM cannot be currently undergoing editing or listing. MEM cannot be either the slave or the master in a checkout relationship. When the ARCHIVE feature is installed the user must have ADM command level security to purge MEM if check control is on.

Use in a Procedure

Return Codes:

OK	Successful.
CK	Member is part of a checkout relationship.
ED	Member is being edited.

LI Member has active LIST session(s).
NF Member not found.
SC Inadequate access level.

PURGE sets the TXM variables to the attributes of MEM.

Examples

Purge member TKRMBR1:

```
=> purge tkrmbr1
```

Purge member XX34:

```
=> pur xx34
```

End the current LIST session and purge the member associated with the session:

```
=> end;purge *
```

PURGED (VSE version)

Use PURGED to delete a VSE sublibrary member.

PURGED may also be entered as PURD.

Required Operands	
MEMD	the name and type of an existing VSE sublibrary member. The currently attached sublibrary (see ATTACHD) will be used unless the library and sublibrary are prefixed to the member name and type (i.e. library.sublib.member.type). If MEMD is specified as an asterisk (*), the "last referenced VSE sublibrary member" will be purged.

PURGED sets the "last referenced VSE sublibrary member". This is typically not useful because no further processing can be done to it.

When Valid

BIM-EDIT must be running in its own partition and under VSE/SP release 2.1 or later.
The user must have DEF access level for the MEMD library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Member not found.
SC	Inadequate access level.

Examples

Delete member TEST.PHASE in the current sublibrary:

=> purged test.phase

Delete member JRNLREC.C in sublibrary USRLIB3.GENL:

```
=> purd usrlib3.genl.jrnlrec.c
```

PURGED (MVS version)

Use PURGED to delete a Partitioned Data Set (PDS) member.

PURGED may also be entered as PURD.

Required Operands	
MEMD	the name of the PDS member. The currently attached PDS (see ATTACHD) will be used unless the PDS name is prefixed to the member name, that is, pdsname.member or pdsname(member). If MEMD is specified as an asterisk (*), the "last referenced PDS member" will be purged.

PURGED sets the "last referenced PDS member". This is typically not useful because no further processing can be done to it.

When Valid

The user must have DEF access level for the MEMD library.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Member not found.
SC	Inadequate access level.

Examples

Delete member TEST2 in the current PDS:

```
=> purged test2
```

Delete member JRNLRD in PDS USERLIB3.GENL.C:

```
=> purd userlib3.genl.c.jrnld
```

Delete member PROG428 in PDS SYSLIB.PROBLEM:

```
=> purged syslib.problem(prog428)
```

PURGEL

Use PURGEL to delete an empty library. (PURGE or RENAME must first be used to remove all members from the library.)

PURGEL may also be entered as PURL.

Required Operands	
LIB	is the library to be purged.

PURGEL can also be invoked by entering the P command in the LCA area of a LIBRARYL display.

The library must be empty to be purged, that is, there must not be any members in the library.

When Valid

The user must have DEFL access level for the library prefix of the LIB library (the part of the library name preceding an imbedded period). If LIB has no prefix, the user must have system-wide DEFL access.

Use in a Procedure

Return Codes:

OK	Successful.
CH	Library has member(s).
NF	Library not found.
SC	Inadequate access level.

PURGEL sets the TXL variables to the attributes of LIB.

Example

Purge library 4321:

=> purgel 4321

PURGEP (VSE version)

Use PURGEP to delete a POWER job entry.

PURGEP may also be entered as PURP or PP.

Optional Operands	
PRM1-4	<p>select the POWER job entry to be deleted. The four operands allow you to specify the queue (RDR,LST,PUN,XMT), the job name, the job number, and the segment number. They can be specified in any order, except that segment, if specified, must follow job number. The following indicates which entry will be selected based on the operands specified:</p> <p>queue, job name, job number, segment</p> <p>The specified queue will be searched for the entry for the specified name, number, and segment.</p> <p>queue, job name</p> <p>The specified queue will be searched for the first entry for the specified name.</p> <p>queue, job number</p> <p>The specified queue will be searched for the first entry for the specified number, without regard to the job name.</p> <p>job number</p> <p>job name</p> <p>The LST queue will be searched for the first entry for the specified number or name.</p> <p>queue</p> <p>The specified queue will be searched for the first entry for the job name and number of the "last referenced POWER job entry"</p> <p>If PRM1-4 is not specified at all, the "last referenced POWER job entry" will be deleted.</p>

PURGEP can also be invoked by entering the P command in the LCA area of a LIBRARYP display. The DISCARD command can be used to end a LISTP session and purge the queue entry associated with it.

PURGEP sets the "last referenced POWER job entry". This is typically not useful because no further processing can be done to the entry.

When Valid

Check with your System Administrator to determine the restrictions on access to POWER job entries at your site. The exit routine distributed with BIM-EDIT provides

several restriction schemes your System Administrator can select among. (Use of and changes to the BIXPWQA exit routine are described in the System Reference Manual.)

Use in a Procedure

Return Codes:

OK	Successful.
NF	Job entry not found.
PW	Command rejected by POWER.
*	Codes as defined by the BIXPWQA exit routine.

PURGE sets the PWR variables to the attributes of the POWER job entry purged.

Examples

Purge reader queue entry with name ASM500, job number 3245:

```
=> purgep rdr,asm500,3245
```

Purge list queue entry with name AX3526:

```
=> pp ax3526
```

Purge list queue entry with job number 2164:

```
=> purgep lst,2164
```

PURGEP (MVS version)

Use PURGEP to delete a JES job or data sets.

PURGEP may also be entered as PURP or PP.

Optional Operands	
QUEUE	specifies the JES queue. QUEUE may be specified as "I" for the input queue, "H" for the held queue or "O" for the output queue.
JOB	selects the JES job to be deleted. If JOB contains letters, it specifies a jobname label from the Job Control Language JOB statement. If JOB is entirely digits, it specifies a job number assigned by JES. If not specified, the job from "last referenced JES data sets" will be used.
GROUP	<p>selects the data sets to be deleted within the specified job. To delete a JES job (as opposed to data sets), do not specify GROUP.</p> <p>If GROUP is a single letter or digit, it specifies all data sets in the GROUP class in the JES "hold queue". If GROUP is any other multiple character value, it specifies all data sets in the GROUP outgrp in the JES "output queue".</p> <p>An outgrp may be composed of three parts separated by periods in the form "name.qualifier.copy" where "name" is the JES or user-specified name, qualifier is a JES-assigned number, and copy is the copy number. All data sets matching a partial outgrp specification (for example, "name" only) will be deleted. To distinguish an outgrp with a one-character name from a class, follow the name with a period.</p> <p>If neither JOB or GROUP is specified, the "last referenced data sets" are deleted. (If "last referenced data sets" was specified with a DSET operand, PURGEP will be rejected because JES provides no facility to delete by data set.)</p>

To delete an entire JES job, specify the JOB operand but do not specify the GROUP operand. If the job has not completed processing, it will be cancelled before all data sets are deleted.

PURGEP operates on all data sets within the job that meet the selection.

PURGEP can also be invoked by entering the P command in the LCA area of a LIBRARYP display. The DISCARD command can be used to end a LISTP session and purge the JES data sets associated with it.

PURGEP sets "last referenced JES data sets". This is typically not useful because no further processing can be done to the entry.

When Valid

Check with your System Administrator to determine the restrictions on access to JES data sets at your site. The exit routine distributed with BIM-EDIT provides several

restriction schemes your System Administrator can select among. (Use of and changes to the BIXPWQA exit routine are described in the System Reference Manual.)

Use in a Procedure

Return Codes:

OK	Successful.
NF	JES job or data sets not found.
*	Codes as defined by the BIXPWQA exit routine.

PURGE sets the JCT, JQE, and PDB variables to the attributes of the JES job or data sets purged. If GROUP is an outgrp, the JOE variables are also set.

Examples

Purge all data sets for job ASM500:

```
=> purgep asm500
```

Purge class G hold queue data sets for job 3526:

```
=> pp 3526,g
```

Purge outgrp 3.1 output queue data sets for the last referenced job:

```
=> purgep group=3.1
```


PURGEQ

Use PURGEQ to delete a mail message.

PURGEQ may also be entered as PURQ.

Required Operands	
ID	is a five digit number assigned to the message.

PURGEQ can also be invoked by entering the P command in the LCA area of a LIBRARYQ display. The DISCARD command can be used to end a mail display session and purge the message associated with it.

The message can reside in the incoming-mail queue, the mail-set-aside queue, or the outgoing-mail queue.

Use in a Procedure

Return Codes:

OK	Successful.
NF	Message not found.

Example

Purge message 278:

```
=> purgeq 278
```

QUALIFY

Use QUALIFY to display all lines in the current session that contain a specified string in a specified column range. When QUALIFY is invoked online, it creates a DISP session for the results.

QUALIFY may also be entered as QUAL.

Optional Operands	
STR	is the string to search for. Up to 72 characters may be specified. If the value of STR begins with the backslash character (\), the search will use Extended Search Pattern rules, as described below. If STR is not specified, the value for STR and any other operands not specified will be inherited from previous search commands, also as described below.
FMT	specifies the format of the created display. Enter "number" or "num" if you want displayed lines to be prefixed with a line number. If FMT is not specified, lines will appear exactly as they appear in the session.
ZONE	is the column range for the search. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to limit the search to columns 10 to 20, specify ZONE as "10-20". ZONE=5 is the same as ZONE=5-5, ZONE=5-* is the same as ZONE=5-253 and ZONE=-5 is the same as ZONE=1-5. If ZONE and STR are not specified, the ZONE specification is inherited from previous search commands, as described below. If ZONE, after any inheriting, is not specified, the current session zone is searched (see the ZONE operand of the DEFINE, ALTER, and SESS commands.)
CASE	specifies whether upper/lower case should be considered when determining matches. If specified as "U", case is ignored ("the" matches "The"). If specified as "M", both case and letters must match. If CASE and STR are not specified, the CASE specification is inherited from previous search commands, as described below. If CASE, after any inheriting, is not specified, "U" is assumed.

Changes made to the text of the session created by QUALIFY will not affect the session the QUALIFY is based on. To make changes to the session the QUALIFY was based on, you must ROTATE to that session. The FMT=NUMBER option is provided for use in the POSITION command after you have ROTATED.

QUALIFY is useful with all session types. For example, LISTP sessions can be searched (for compilation error messages or other tell-tale results) and LIBRARY or LIBRARYP displays can be searched. It is valid, and often useful, to use QUALIFY on the session resulting from a QUALIFY.

LCA commands can be applied to lines of the session created by QUALIFY just as they could from an original display session. This is particularly useful for QUALIFY displays based on a LIBRARY, LIBRARYD, or LIBRARYP display.

All of the screen attributes of the original session are maintained in the newly created session.

If your terminal is running under control of SNA, you may interrupt a long QUALIFY command by pressing the ATTN key.

If the value of STR begins with a backslash (\), BIM-EDIT treats STR as an Extended Search Pattern, with certain characters having special meanings:

- ? matches any single character
- * matches any sequence of zero or more characters
- @ matches zero or more instances of the character that follows the @
- ! matches the beginning of the zone
- < matches the beginning of the zone or any non-alphanumeric character
- > matches the end of the zone or any non-alphanumeric character
- | separates two patterns, EITHER of which may match for a successful search
- + separates two patterns, BOTH of which must match for a successful search
- ~ (NOT) reverses line-by-line success/failure of the pattern following it
- % marks the position on the line where the match is considered to occur
- \ suppresses the special meaning of the character that follows the \

To QUALIFY for a text string at a specific column (like FIND), you can use Extended Search Patterns like

```
=> qualify '\!string' zone=10-*
```

To QUALIFY for the absence of a text string at a particular column, (like NFIND), you can use Extended Search Patterns like

```
=> qualify '\~!string' zone=10-*
```

See Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual for more information about Extended Search Patterns.

If STR is not specified on the command, BIM-EDIT uses as default operand values those from previous search commands. Using this feature, you can easily request searches which are similar or identical to the last one. For example, after a LOCATE command which found more instances than you were expecting, you need only enter

```
=> qualify
```

to create a session showing all instances. To achieve this, BIM-EDIT stores a previous value for the STR, COL, CASE, and ZONE operands of the search commands (EXAMINE, FIND, FINDUP, LOCATE, LOCATEU, NFIND, NFINDUP, QUALIFY, or SCAN). These previous values will be inherited by operands omitted from a search command which does not specify STR. Whenever a search command specifies STR, the previous values are set to those specified on that command or "not specified".

Use in a Procedure

Return Codes:

OK Successful.
 AT Command interrupted by ATTN key.
 NF String not found.

If QUALIFY is invoked in batch utility, the display destination is controlled by predefined variable SIBOUTPT.

Example

Display all lines containing the string "1.6":

LIBRARY session before QUALIFY command

```
=> qualify 1.6
DISP -> lib dfh.                      SESS=A 1( 1) LINE=    0(   95)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==== -- TOP OF DISPLAY --
*==== LIBRARY=DFH
*==== -----
*==== MEMBER          TYPE          TITLE
*==== -----
*==== ASGN3340         JCL
*==== BIMGMM17         ASM          CICS 1.7 BIM GOOD MORNING MESSAGE
*==== BIMLIBTB         ASM          VSE/SP 3.1 BIMLIBTB PARAMETERS
*==== CICSBCPD         JCL
*==== CICSICCF         JCL
*==== CICSICCF-ORIG    JCL
*==== CICSPROD         JCL          CICS 1.7 PRODUCTION STARTUP
*==== CICSPROD-SPCKD   JCL
*==== CICSPROD-VSE3    JCL          CICSPROD 1.6 JOB STREAM
*==== CICSPROD-16      JCL
*==== CICSPRV3-CKD     JCL          CICSPROD 1.6 JOB STREAM
```

QUALIFY session

```
=>
DISP -> qualify 1.6                      SESS=A 2( 2) LINE=    0(    6)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==== -- TOP OF DISPLAY --
*==== CICSPROD-VSE3    JCL          CICSPROD 1.6 JOB STREAM
*==== CICSPRV3-CKD     JCL          CICSPROD 1.6 JOB STREAM
*==== CICSTEST-SPCKD   JCL          CICSTEST 1.6 JOB STREAM
*==== CICSTEST-VSE3    JCL          CICSTEST 1.6 JOB STREAM
*==== CICSTEST-16      JCL          CICSTEST 1.6 JOB STREAM
*==== CICS16RS         JCL          CICS 1.6 SYSTEM TAPE RESTORE
*==== -- END OF DISPLAY --
```

Display all lines containing the word "Link", with their line numbers:

Session before QUALIFY command

```
=> qual \<Link>,num
EDIT -> RND.COMMSTDS                      SESS=A 1( 1) LINE=    0(   19)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF MEMBER --
*====* ISO 8802-2      Logical Link Control
*====* ISO 9314-2      FDDI Token Ring Media Access Control
*====* ANSI X3.167     LDDI Data Link Protocol and Network Specific Sub-Layer
*====* ISO 8802-3      CSMA/CD Media Access Control
*====* ISO 8802-4      Token Bus Media Access Control
*====* IEEE 802.1d     MAC Bridges
*====* ISO 8802-5      Token Ring Media Access Control
*====* ISO 7776        Multilink Procedure (MLP)
*====* CCITT X.25      Multilink Procedure (MLP)
*====* ISO 8802-7      Slotted Ring Access Method
```

QUALIFY session

```
=>
DISP -> qual \<Link>,num                      SESS=A 2( 2) LINE=    0(    5)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----|-----
-- TOP OF DISPLAY --
   1 ISO 8802-2      Logical Link Control
   3 ANSI X3.167     LDDI Data Link Protocol and Network Specific Sub-Layer
  11 ISO 7776        Single Link Procedure LAP/LAPB
  12 CCITT X.25      Single Link Procedure LAP/LAPB
  18 CCITT Q.703     Signalling System No. 7/Signalling Link Protocol
-- END OF DISPLAY --
```

QUIT

Use QUIT to terminate the current session and retain any updates.

QUIT may also be entered as SAVE, Q, or FILE.

QUIT has no operands.

After the transaction the screen will display the session that precedes it in the session chain.

The END command with the NOSAVE option can be used to end an EDIT or EDITD session without saving any changes to the text.

When an EDIT or LIST session is saved, the "last referenced member" is set to the member for the session saved.

On VSE, when a LISTP session is saved, the "last referenced POWER job entry" is set to the POWER job entry for the session saved.

On MVS, when a LISTP session is saved, "last referenced JES job or data sets" is set to the JES data sets for the session saved.

On VSE, when an EDITD or LISTD session is saved, the "last referenced VSE sublibrary member" is set to the member for the session saved.

On MVS, when an EDITD or LISTD session is saved, the "last referenced PDS member" is set to the member for the session saved.

This is useful because there will often be further commands desired (such as PROCESS or PURGEP). But it can be confusing because "last referenced" does not correspond to what is showing on the screen after a QUIT.

Use in a Procedure

QUIT normally returns OK. (See SIBRETC D.)

When an EDIT or LIST session is saved, the TXM variables are set to the attributes of the member for the session saved.

On VSE, when a LISTP session is saved, the PWR variables are set to the attributes of the POWER job entry for the session saved.

On MVS, when a LISTP session is saved, the JQE, JCT, and PDB variables are set to the attributes of the JES data sets for the session saved.

Example

Save the existing session:

```
=> quit
```

REFRESH

Use REFRESH to recreate the current display session.

REFRESH may also be entered as REF.

Optional Operands	
TIME	Time interval seconds for automatic refresh. TIME=N, where N is a number from 3 seconds to 15 seconds.
LIMIT	Limit, in total seconds, for automatic refresh. LIMIT=N, where N is a multiple of 'TIME' up to a maximum of 1800. If this parameter is not provided, the limit is infinite. Pressing an entry key will terminate the automatic refresh.

REFRESH is useful with status display sessions such as those produced by DA, LIBRARY, LIBRARYP, and SHOW. These sessions always display the status as of the time the command was issued. Since subsequent changes to the status do not update the session, it is necessary to reissue the command to see up-to-date status. The REFRESH command simplifies this by providing a single command that will end a display session and reissue the same command that originally created it.

Note: that REFRESH may not have the intended effect if the original command had implied operands whose default values have changed since the command was issued. Also, timed REFRESH is ineffective for entities connected to BIM-EDIT via XPCC or LU6.2 protocols (that includes CICS terminals).

This command is used frequently enough that you'll probably want to assign it to a PF key.

Use in a Procedure

REFRESH normally returns OK. (See SIBRETCDD.)

Example

Refresh the current session:

```
=> refresh
```

Refresh the current session automatically on time interval:

```
=> refresh time=5  
- or -  
=> ref 5
```

RELEASE (VSE version)

Use RELEASE to take a POWER job entry off hold, that is, alter it so it will be automatically processed by POWER.

RELEASE may also be entered as REL.

Optional Operands	
PRM1-4	<p>select the POWER job entry to be altered. The four operands allow you to specify the queue (RDR,LST,PUN,XMT), the job name, the job number, and the segment number. They can be specified in any order, except that segment, if specified, must follow job number. The following indicates which entry will be selected based on the operands specified:</p> <p>queue, job name, job number, segment</p> <p>The specified queue will be searched for the entry for the specified name, number, and segment.</p> <p>queue, job name</p> <p>The specified queue will be searched for the first entry for the specified name.</p> <p>queue, job number</p> <p>The specified queue will be searched for the first entry for the specified number, without regard to the job name.</p> <p>job number job name</p> <p>The LST queue will be searched for the first entry for the specified number or name.</p> <p>queue</p> <p>The specified queue will be searched for the first entry for the job name and number of the "last referenced POWER job entry"</p> <p>If PRM1-4 is not specified at all, the "last referenced POWER job entry" will be used.</p>

RELEASE can also be invoked by entering the R command in the LCA area of a LIBRARYP display.

To place a job entry on hold, use HOLD.

RELEASE sets the "last referenced POWER job entry".

When Valid

Check with your System Administrator to determine the restrictions on access to POWER job entries at your site. The exit routine distributed with BIM-EDIT provides

several restriction schemes your System Administrator can select among. (Use of and changes to the BIXPWQA exit routine are described in the System Reference Manual.)

Use in a Procedure

Return Codes:

OK	Successful.
NF	Job entry not found.
PW	Rejected by POWER.
*	Other codes as set by BIXPWQA exit routine.

RELEASE sets the PWR variables to the attributes of the POWER job entry released.

Examples

Release an entry in the LST queue, identifying it with queue and job name:

```
=> release lst,asm100
```

Release an entry in the RDR queue, identifying it with queue and job number:

```
=> rel rdr,1278
```

Release an entry in the LST queue, identifying it with just the job number:

```
=> release 3426
```

Release the last referenced POWER job entry:

```
=> release
```

RELEASE (MVS version)

Use RELEASE to take a JES job or data sets off hold, that is, alter it so it will be automatically processed by JES. The job or data sets must be in "hold" status.

RELEASE may also be entered as REL.

Optional Operands	
JOB	selects the JES job to be released. If JOB contains letters, it specifies a jobname label from the Job Control Language JOB statement. If JOB is entirely digits, it specifies a job number assigned by JES. If not specified, the job from "last referenced JES data sets" will be used.
GROUP	<p>selects the data sets to be released within the specified job. To release a JES job (as opposed to data sets), do not specify GROUP.</p> <p>If GROUP is a single letter or digit, it specifies all data sets in the GROUP class in the JES "hold queue". If GROUP is any other multiple character value, it specifies all data sets in the GROUP outgrp in the JES "output queue".</p> <p>An outgrp may be composed of three parts separated by periods in the form "name.qualifier.copy" where "name" is the JES or user-specified name, qualifier is a JES-assigned number, and copy is the copy number. All data sets matching a partial outgrp specification (for example, "name" only) will be released. To distinguish an outgrp with a one-character name from a class, follow the name with a period.</p> <p>If neither JOB or GROUP is specified, the "last referenced data sets" are released. (If "last referenced data sets" was specified with a DSET operand, RELEASE will be rejected because JES provides no facility to release by data set.)</p>

To release a JES job (as opposed to data sets) specify the JOB operand but do not specify the GROUP operand. To usefully release a JES job, the job must not have started processing.

RELEASE operates on all data sets within the job that meet the selection.

The relationship between the "output queue" and the "hold queue" is a little confusing. For a data set to be in the "hold queue", HOLD=YES must have been specified or implied by the CLASS on the DD statement that created it and it must never have been RELEASEd or altered to a CLASS which implies HOLD=NO. When a HOLD command is issued for a data set that is in the "output queue", the data set is placed on hold status but remains in the "output queue", and must therefore be accessed by outgrp rather than by class.

RELEASE can also be invoked by entering the R command in the LCA area of a LIBRARYP display.

To place a job or data sets on hold, use HOLD.

RELEASE sets the "last referenced JES data sets".

When Valid

Check with your System Administrator to determine the restrictions on access to JES data sets at your site. The exit routine distributed with BIM-EDIT provides several restriction schemes your System Administrator can select among. (Use of and changes to the BIXPWQA exit routine are described in the System Reference Manual.)

Use in a Procedure

Return Codes:

OK	Successful.
NF	JES job or data sets not found.
*	Other codes as set by BIXPWQA exit routine.

RELEASE sets the JQE, JCT, and PDB variables to the attributes of the JES job or data sets released. If GROUP is an outgrp, the JOE variables are also set.

Examples

Release all data sets for job ASM100:

```
=> release asm100
```

Release hold queue class G data sets for job 7823

```
=> rel 7823,G
```

Release the last referenced job:

```
=> release
```

RENAME

Use RENAME to change the name of a member. RENAME is also used to move a member from one library to another.

RENAME may also be entered as REN.

Required Operands	
SRC	is the existing member name.
DEST	is the new member name. A member having the destination name must not already exist.
PSWD	If the member is password protected, the password must be entered. The password may be up to 8 characters in length consisting of any characters. The renamed member remains password protected.
REPL	is used to specify whether the DEST member is to be replaced if a member by this name already exists. The default is "NO". If REPL=YES is specified and the DEST member does not exist the RENAME will be completed but the completion message will display the fact that the DEST member was not found. It is not possible to replace the following: <ul style="list-style-type: none">• a master member (i.e. check status is on)• a member that is part of a checkout relationship• a member that is password protected

The FRENAME command may provide a more convenient way to rename a member.

If an asterisk (*) is entered as part of DEST, the asterisk is replaced with the corresponding part of SRC member name. This offers an easy way of moving a member from one library to another or adding a suffix or prefix to a name without keying the member name twice (see Examples).

RENAME sets the "last referenced member" to DEST.

When Valid

The user must have DEF access level for both the SRC library and the DEST library. SRC cannot be currently undergoing editing or listing. SRC cannot be either the slave or the master in a checkout relationship.

Use in a Procedure

Return Codes:

OK	Successful.
CK	SRC member is part of a checkout relationship.
DP	DEST member already exists.
ED	SRC member is being edited.
LI	SRC member has active LIST session(s).
NF	One of the following:

- SRC member not found.
- DEST library not found.

RENAME sets the TXM variables to the attributes of DEST.

Examples

Rename SYRINT1 to TKRINT1:

```
=> rename syrint1,tkrint1
```

Move member XMRL121 to library 4216:

```
=> ren xmrl121,4216.xmrl121      -or-
=> ren xmrl121,4216.*
```

RENAMED (VSE version)

Use RENAMED to change the name of a VSE member. RENAMED is also used to move a VSE member from one VSE library to another.

RENAMED may also be entered as REND.

Required Operands	
SRC	is an already existing member. The currently attached sublibrary (see ATTACHD) will be used unless the library and sublibrary are prefixed to the member name (i.e. lib.sublib.member.type).
DEST	is the new member name. A member of this name must not already exist unless the REPL operand is specified as YES. The currently attached sublibrary (see ATTACHD) will be used unless the library and sublibrary are prefixed to the member name (i.e. lib.sublib.member.type).

Optional Operands	
REPL	If the member name already exists in the destination library, a value of "YES" may be entered to cause the member to be overwritten. If omitted, the default is "NO".

When Valid

The user must have DEF access level for both the SRC library and the DEST library. SRC cannot be currently undergoing editing or listing.

Use in a Procedure

Return Codes:

- OK Successful.
- DP DEST member already exists.
- ED SRC member is being edited.
- LI SRC member has active LIST session(s).
- NF One of the following:
 - SRC member not found.
 - DEST library not found.

Examples

Rename SYRINT1.A to TKRINT1.A:

```
=> renamed syrint1.a,tkrint1.a
```

Move member XMRL121.a to library BACKUP:

```
=> rend xmrl121.a,backup.xmrl121
```

RENAMED (MVS version)

Use RENAMED to change the name of a PDS member. RENAMED is also used to move a PDS member from one PDS to another.

RENAMED may also be entered as REND.

Required Operands	
SRC	is an already existing PDS member. The currently attached PDS (see ATTACHD) will be used unless the PDS name is prefixed to the member name, that is, pds.member or pds(member).
DEST	is the new member name. A member of this name must not already exist. The currently attached PDS (see ATTACHD) will be used unless the PDS name is prefixed to the member name, that is, pds.member or pds(member).

When Valid

The user must have DEF access level for both the SRC library and the DEST library. SRC cannot be currently undergoing editing or listing.

Use in a Procedure

Return Codes:

- | | |
|----|---|
| OK | Successful. |
| DP | DEST member already exists. |
| ED | SRC member is being edited. |
| LI | SRC member has active LIST session(s). |
| NF | One of the following: <ul style="list-style-type: none">• SRC member not found.• DEST library not found. |

Examples

Rename OLDMEM to NEWMEM in the current PDS:

```
=> renamed oldmem newmem
```

Rename member PYRLMNTH from the current PDS to the PRODLIB.PAYROLL PDS:

```
=> renamed pyrlmnth prodlib.payroll.pyrlmnth
```

RESEQ

Use RESEQ to replace current session text by sequence numbers at a defined column range on all lines.

RESEQ may also be entered as RESE, RES, or RENUM.

Optional Operands	
BASE	specifies the starting sequence number. If BASE is not specified, the starting sequence number will be the value for INCR.
INCR	specifies the increment amount, that is, the difference between the numbers on two successive lines. If INCR is not specified, a value of 100 is assumed.
AUDIT	specifies whether auditing should be on during the resequence. Enter OFF or NO to turn auditing off during the resequence request. A one line audit comment is written to the audit trail indicating that the AUDIT=OFF option was use when the RESEQ was done. If AUDIT is not specified it will default to the member audit option. The default is YES or ON.

The column range for the sequence numbers must have been previously specified through the SEQ operand of the DEFINE, ALTER or SESS command. It cannot be wider than 8 characters.

Certain lines are not overlaid with sequence numbers. Any line beginning with a right parenthesis ")", the BIM-EDIT TRAP character, is bypassed. Also, any line beginning with "/INCL" is bypassed.

The text on all lines to be updated must be blank or numeric in the column range. If it finds non-numeric data, RESEQ's only action is to position the session to the line which contains the non-numeric data.

Lines that are updated by RESEQ will be written to the member's audit trail (if member auditing is active and AUDIT=NO has not been specified) but will not have their date stamps updated. See the BIM-EDIT User Reference Manual, Chapter 6, Advanced Techniques, for more discussion of member auditing and member stamping.

Use in a Procedure

RESEQ normally returns OK. (See SIBRETC D.)

Examples

Assuming that SEQ=1-6, assign sequence numbers, starting at 10000 and incrementing by 1000:

EDIT session before RESEQ command

```
=> reseq 10000,1000
EDIT  ED51A.BIMAPCB                      SESS=A 2( 2) LINE=      0(    68)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF MEMBER --
*====*      IDENTIFICATION DIVISION.
*====*      PROGRAM-ID. BIMAPCB.
*====*      *****
*====*      * BIMAPCB - SAMPLE USE OF BIM-EDIT APPLICATION INTERFACE
*====*      *
```

EDIT session after RESEQ command

```
=>
## RESEQUENCE COMPLETE ##
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF MEMBER --
*====* 010000 IDENTIFICATION DIVISION.
*====* 011000 PROGRAM-ID. BIMAPCB.
*====* 012000*****
*====* 013000* BIMAPCB - SAMPLE USE OF BIM-EDIT APPLICATION INTERFACE
*====* 014000*
```

RESET

Use RESET to forget a pending line control area (LCA) bracket command.

RESET has no operands.

In practice, RESET is used when an LCA bracket command, (for example, delete - DD), is entered and the session is repositioned so that the LCA command is no longer visible. If it becomes desirable to cancel the pending LCA command, RESET can be used. The only other way of removing the pending LCA command is to reposition the session so that the LCA command is visible and then remove the command by blanking out the LCA where the command is displayed.

Use in a Procedure

RESET normally returns OK. (See SIBRETCDD.)

Example

Remove a pending LCA command:

```
=> reset
```

ROTATE

Use ROTATE to select which of your sessions will be the current session.

ROTATE may also be entered as ROT.

Optional Operands	
OPT	<p>Values for OPT may be:</p> <p>+ or F indicating the next session on the session chain.</p> <p>- or B indicating the previous session on the session chain.</p> <p>sess specifying a session number.</p> <p>If OPT is not specified, the next session on the session chain will be displayed.</p>

Sessions are maintained on a chain. Any session may be displayed at any time by use of the ROTATE command. Rotation is useful when editing or comparing multiple members or when editing a member in response to a LISTP session of a listing. You may find it desirable to set PF keys to "ROT +" and "ROT -". ROTATE may be used in conjunction with SCREEN SPLIT to show different sessions in the upper and lower logical screens.

The field "SESS=" on the information line shows two values. The first identifies which session is currently displayed. The second value, within parentheses, is the total number of active sessions. For example, "SESS=A 1(4)" denotes that the current session is the first of four sessions.

If the session you rotate to is a LIST or EDIT session, the "last referenced member" entry (variables TXMLIB, TXMMEM and TXMPATH) will be set. If the session you rotate to is a LISTP session, "last referenced POWER job entry" (variables PWRJNAME, PWRJNUM or PWRTYPE) or "last referenced JES data sets" (all the JQE variables) will be set. If the session you rotate to is a LISTD or EDITD session, "last referenced VSE sublibrary member" or "last referenced MVS PDS member" (variable SIBLSPDS) will be set.

Use in a Procedure

ROTATE normally returns OK. (See SIBRETCD.)

A syntax peculiarity occurs if "ROTATE -" is used in a procedure. The space dash (" -") that terminates the command will be taken as a line continuation, usually leading to a command error. The simplest way to avoid the error is to use "ROTATE '-".

Examples

Rotate to the next session in chain:

```
=> rotate
```

Rotate to the previous session in chain:

```
=> rot -
```

Rotate to the first session:

```
=> rotate 1
```

SAVE

Use SAVE to terminate the current session and retain any updates.

SAVE may also be entered as QUIT, Q, or FILE.

SAVE has no operands.

After the transaction the screen will display the session that precedes it in the session chain.

The END command with the NOSAVE option can be used to end an EDIT or EDITD session without saving any changes to the text.

When an EDIT or LIST session is ended, the "last referenced member" entry (variables TXMLIB, TXMMEM and TXMPATH) are set to the member for the session ended.

When a LISTP session is ended, the "last referenced POWER job entry" (variables PWRJNAME, PWRJNUM and PWRTYPE) or "last referenced JES data sets" (all of the JQE variables) is set to the entry for the session ended.

When an EDITD or LISTD session is ended, the "last referenced VSE sub- library member" or "last referenced MVS PDS member" (variable SIBLSPDS) is set to the entry for the session ended.

This is useful because there will often be further commands desired (such as PROCESS or PURGEP). But it can be confusing because "last referenced" does not correspond to what is showing on the screen after a SAVE.

Use in a Procedure

SAVE normally returns OK. (See SIBRETCD.)

Example

Save the existing session:


```
=> save
```

SCAN

Use SCAN to search multiple members for a specified string of characters. SCAN will display all occurrences of the string within the members. When SCAN is invoked online, it creates a DISP session for the results. If SCAN encounters a password protected member it will be skipped unless the user has ADM level access.

Optional Operands	
MEM	<p>is the library and the initial characters of the members to be searched. If no library is specified (i.e., MEM does not contain a period) members in the currently attached library are searched. If the library is specified but the member is not (i.e., MEM ends in a period), all members in the specified library are searched. If MEM is not specified, all members in the current library are searched. MEM can contain * and ? characters, which will match as described below in the Extended Search Pattern rules. If the library name part of MEM has * or ? characters, you can search members in more than one library.</p> <p>MEM can be specified as '/' to indicate that the current session contains a list of members to be searched. The current session must be the format generated by a LIBRARY command, or the output of a prior SCAN command.</p>
STR	<p>is the string to search for. Up to 72 characters may be specified. If the value of STR begins with the backslash character (\), the search will use Extended Search Pattern rules, as described below. If STR is not specified, the value for STR, ZONE, and CASE will be inherited from previous search commands, also as described below.</p>
FMT	<p>is the format of the member line display. The following are provided:</p> <p>NAME displays member name only.</p> <p>DET displays member name, type, attributes, size, and dates.</p> <p>FULL displays the DET information plus user ID and title.</p> <p>CHECK displays information for checkout/checkin management.</p> <p>If FMT is not specified, the default format shows NAME, TYPE, and TITLE on a single line. See the LIBRARY command for more information about FMT.</p>
TYPE	<p>If TYPE is specified, only members which match the specified TYPE are searched. If the value of TYPE begins with a backslash character (\), matching will use Extended Search Pattern rules, as described below. See the DEFINE command for information</p>

	about TYPE.
--	-------------



Optional Operands (continued)	
ATTR	If ATTR is specified, only members which match the specified ATTR are searched. If the value of ATTR begins with a backslash character (\), matching will use Extended Search Pattern rules, as described below. See the DEFINE command for information about ATTR.
CKUSER	If CKUSER is specified, only members that have been checked out to the CKUSER user are searched. If CKUSER is specified as an asterisk (*), all members that have been checked out are displayed. If the value of CKUSER begins with a backslash character (\), matching will use Extended Search Pattern rules, as described below.
TITLE	If TITLE is specified, only members that contain its value in their TITLES are searched. If the value of TITLE begins with a backslash character (\), matching will use Extended Search Pattern rules, as described below.
ZONE	is the column range for the search. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to limit the search to columns 10 to 20, specify ZONE as "10-20". ZONE=5 is the same as ZONE=5-5, ZONE=5-* is the same as ZONE=5-253 and ZONE=-5 is the same as ZONE=1-5. If ZONE and STR are not specified, the ZONE specification is inherited from previous search commands, as described below. If ZONE, after any inheriting, is not specified, 1-253 is used as the default.
CASE	specifies whether upper/lower case should be considered when determining matches. If specified as "U", case is ignored ("the" matches "The"). If specified as "M", both case and letters must match. If CASE is not specified, "U" is assumed.
TEXT	specifies how the matching text lines should be displayed. If specified as "YES", text lines are displayed in addition to member lines. If specified as "NUM", line numbers are displayed on a separate line before the text lines. If specified as "NO", only member lines are displayed. (The result will look like a LIBRARY display.) If TEXT is not specified, "YES" is assumed.
SEARCH	specifies whether the search should be made in text lines or the audit trail of the matching members. If specified as 'TEXT', the text lines are searched. If specified as 'AUDIT', the audit trail lines are searched. If omitted, 'TEXT' is assumed.

If your terminal is running under control of SNA, you may interrupt a long SCAN command by pressing the ATTN key.

When SCAN is executed from the online command line, a status message will be displayed periodically in the line-2 message area showing the progress of the scan. The

most recent member scanned is shown along with counts of the number of libraries and members that have been scanned, and the total number of members that have been matched.

If the value of STR begins with a backslash (\), BIM-EDIT treats STR as an Extended Search Pattern, with certain characters having special meanings:

- ? matches any single character
- * matches any sequence of zero or more characters
- @ matches zero or more instances of the character that follows the @
- ! matches the beginning of the zone
- < matches the beginning of the zone or any non-alphanumeric character
- > matches the end of the zone or any non-alphanumeric character
- | separates two patterns, EITHER of which may match for a successful search
- + separates two patterns, BOTH of which must match for a successful search
- ~ (NOT) reverses line-by-line success/failure of the pattern following it
- % marks the position on the line where the match is considered to occur
- \ suppresses the special meaning of the character that follows the \

See Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual for more information about Extended Search Patterns. The following LCA commands can be used on the member name lines of the SCAN display to process the member indicated:

E	EDIT	Q	FALTER
L	LIST	S	PROCESS
P	PURGE		

See Chapter 5, LCA Commands, in the BIM-EDIT User Reference Manual for more information.

The SCAN display member name lines have control information to the right of column 135 for use by LCA commands. If you use SCAN display lines in another context, you may need to remove the control information using the BLANK or KEEP commands.

If STR is not specified on the command, BIM-EDIT uses as default operand values those from previous search commands. Using this feature, you can easily request searches which are similar or identical to the last one. For example, after a LOCATE command which failed because the current session does not contain the string, you need only enter

```
=> scan
```

to scan the entire current library for the string. To achieve this, BIM-EDIT stores a previous value for the STR, COL, CASE, and ZONE operands of the search commands (EXAMINE, FIND, FINDUP, LOCATE, LOCATEU, NFIND, NFINDUP, QUALIFY, or SCAN). These previous values will be inherited by operands omitted from a search command which does not specify STR. Whenever a search command specifies STR, the previous values are set to those specified on that command or "not specified".

When Valid

The user must have LIST access level for the libraries searched.

Use in a Procedure

Return Codes:

OK	Successful.
AT	Command interrupted by ATTN key.
NF	Library not found.
SC	Inadequate access level.

If SCAN is invoked in batch utility, the display destination is controlled by predefined variable SIBOUTPT.

Example

Report all occurrences of the string "USING" within the library "DFH":

```
=> scan dfh.,using
```

DISP session after SCAN command

```
=>
DISP -> scan dfh.,using                      SESS=A 3( 3) LINE=    0( 435)
-----1-----2-----3-----4-----5-----6-----7--
*==* -- TOP OF DISPLAY --
*==* LIBRARY=DFH, SEARCH STRING=USING
*==* -----
*==* MEMBER          TYPE          TITLE
*==* -----
*==* BIMGMM23        ASM          CICS 2.3 BIM GOOD MORNING MESSAGE
*==*
*==*          USING BIMGMM,R9                      @BM10954 0
*==*
*==* DFHGMM21        ASM
*==*
*==*          USING DFHLFS,R9                      @BM10954 0
*==*
*==* DFHSRPS2        JCL          OLTD 1.3 DFHSRP STAGE 2
*==*
*==* *          EXITS =      RETURNS TO DOS/V5 USING 'EXIT PC' MACRO          * 0
```

SCREEN

Use SCREEN to set session display modes. Modes can be set on, set off, or "toggled" to the opposite setting.

SCREEN may also be entered as SCR.

Optional Operands	
MODE	<p>Specify one of the following modes:</p> <p>ALT is alternate mode. Certain IBM or compatible terminals can assume two different screen sizes (for example, 132 columns wide or 43 lines deep). When alternate mode is on, the alternate screen size is used. If you change alternate mode while you are in a session, the mode will be changed for that session only. If you change alternate mode when you are not in any sessions, the mode used to start new sessions will be set. Check with your System Administrator for your terminal's capabilities.</p> <p>AUD is audit display mode. It may also be specified as AUDIT. When audit display mode is on, the member audit trail is displayed rather than the member text.</p> <p>BEF is before mode. When before mode is on, a number of text lines indicated by the VAL operand are displayed above the scale line. (The current line is always the one immediately below the scale line.)</p> <p>BEFDF sets the default before mode to use for your sessions.</p> <p>HEX is hex mode. When hex mode is on, lines are displayed in a three line EBCDIC/hexadecimal format. The LCA is not displayed. See Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual for discussion.</p> <p>LCA is line control area mode. When LCA mode is on, the line control area is shown on the screen on the side indicated by the VAL operand. When it is off, no LCA area shows, just the text of the session. When an EDIT session or a LIST session is created, LCA mode is initially on. When a LISTP session is created, LCA mode is initially off. DISP sessions vary depending on the command that created them. LCA mode may be changed at anytime.</p>

Optional Operands (continued)	
	<p>SPLIT is split-screen mode. It may also be specified as SP. When split-screen is on, two logical screens will be displayed. The line position of the split point will be determined by the VAL operand.</p> <p>The logical screens may show parts of the same session or different sessions. The < command may be used to select which logical screen the command line affects. The SWAP command may be used to exchange the top and bottom logical screens. See Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual for discussion.</p> <p>STAMP is stamp mode. When stamp mode is on, stamp information is displayed in the LCA area in the format indicated by the VAL operand. (It may be necessary to erase the LCA to use some LCA commands in this mode.)</p> <p>If MODE is not specified, SPLIT is assumed.</p>
OPER	is the mode setting desired. Specify "ON" to set the mode on, "OFF" to set it off. Specify "TOG" or "T" to toggle the mode to its opposite setting (for example, if the mode is currently on, TOG will set it off). If OPER is unspecified, TOG is assumed.
VAL	<p>is used with BEF, LCA, SPLIT, and STAMP modes. Its meaning depends on the value of MODE:</p> <p>BEF VAL specifies the number of lines to display above the scale line and current line. If VAL is not specified, four lines will precede the scale line.</p> <p>LCA VAL specifies where to place the LCA. "LEFT" places the LCA on the left side of the text display area and "RIGHT" places the LCA on the right side. If VAL is not specified, the LCA will be placed on the side indicated by the value of the predefined variable SIBDFLCA ("L" for the left side and "R" for the right side).</p> <p>SP VAL specifies the line to split at. Each logical screen requires at least 6 lines. VAL can thus never be less than 7. And for a 3270 model 2 with 24 lines, VAL cannot be greater than 19. If VAL is not specified, the screen will be split midway.</p>

Optional Operands (continued)															
	STAMP	VAL specifies the format of the displayed stamp information. If a line has been modified or added while member stamping was active, the following stamp information is available: t type of update (A=add, M=modify) yy year update made mm month update made dd day update made uuuu first 4 characters of user who made update Stamp information is displayed formatted as follows, depending upon the value of VAL and the date format (specified by your System Administrator in variable MMPDTCTL): <table><tr><th>VAL</th><th>U.S.A. Format</th><th>International Format</th></tr><tr><td>1</td><td>mmddt</td><td>ddmmt</td></tr><tr><td>2</td><td>mmyyt</td><td>mmyyt</td></tr><tr><td>3</td><td>uuuut</td><td>uuuut</td></tr></table> If VAL is not specified, format 1 is used.		VAL	U.S.A. Format	International Format	1	mmddt	ddmmt	2	mmyyt	mmyyt	3	uuuut	uuuut
VAL	U.S.A. Format	International Format													
1	mmddt	ddmmt													
2	mmyyt	mmyyt													
3	uuuut	uuuut													

Use in a Procedure

Return Codes:

- OK Successful.
- AU Auditing has not been set on for this member (response to SCREEN AUD).

Examples of SCREEN ALT

Use the alternate screen size:

```
=> screen alt,on
```

Use the normal screen size:

```
=> scr alt,off
```

Use the other screen size:

```
=> screen alt
```

Normal Screen (On a model-5 terminal)

```
=>
DISP -> lib $sys.help. full          SESS=A 1( 1) LINE=      0( 814)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----
*==== -- TOP OF DISPLAY --
*==== LIBRARY=$SYS.HELP
*====
*==== MEMBER          TYPE      ATTR      --- LINES ---      DATE -----
*==== TEXT      AUDIT      CREATE      UPDATE
*====
*==== ABENDXIT        TEXT      73      0 05/23/1997
*==== ABX             TEXT      0      0 05/23/1997
*==== ADD             TEXT      54      0 10/23/1997 10/23/1997
*==== ALT             TEXT      0      0 05/23/1997
*==== ALTER           TEXT      128     143 05/23/1997 04/12/1999
*==== ALTERL          TEXT      53      0 05/23/1997
*==== ALTERP          TEXT      112     47 05/23/1997 04/12/1999
*==== ALTERS          TEXT      60      0 05/23/1997
*==== ALTERU          TEXT      106     0 05/23/1997
*==== ALTL            TEXT      0      0 05/23/1997
*==== ALTP            TEXT      0      0 05/23/1997
*==== ALTS            TEXT      0      0 05/23/1997
*==== ALTU            TEXT      0      0 05/23/1997
*==== AP              TEXT      0      0 05/23/1997
*==== ARCHIVE         TEXT      150     0 05/23/1997
```

Alternate Screen (On a model-5 terminal)

```
=>
DISP -> lib $sys.help. full          SESS=A 1( 1) LINE=      0( 814)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----|-----8-----|-----9-----|-----10-----|-----11-----|-----12-----
*==== -- TOP OF DISPLAY --
*==== LIBRARY=$SYS.HELP
*====
*==== MEMBER          TYPE      ATTR      --- LINES ---      DATE -----
*==== TEXT      AUDIT      CREATE      UPDATE      AUD STM  CREATE  UPDATE      TITLE
*==== ST  ST  USER      USER
*====
*==== ABENDXIT        TEXT      73      0 05/23/1997      ON  OFF $SYS
*==== ABX             TEXT      0      0 05/23/1997      ON  OFF $SYS
*==== ADD             TEXT      54      0 10/23/1997 10/23/1997  ON  OFF LSL      LSL      -> ABENDXIT
*==== ALT             TEXT      0      0 05/23/1997      ON  OFF $SYS
*==== ALTER           TEXT      128     143 05/23/1997 04/12/1999  ON  OFF $SYS      PG      -> ALTER
*==== ALTERL          TEXT      53      0 05/23/1997      ON  OFF $SYS
*==== ALTERP          TEXT      112     47 05/23/1997 04/12/1999  ON  OFF $SYS      FKE
*==== ALTERS          TEXT      60      0 05/23/1997      ON  OFF $SYS
*==== ALTERU          TEXT      106     0 05/23/1997      ON  OFF $SYS
*==== ALTL            TEXT      0      0 05/23/1997      ON  OFF $SYS
*==== ALTP            TEXT      0      0 05/23/1997      ON  OFF $SYS
*==== ALTS            TEXT      0      0 05/23/1997      ON  OFF $SYS
*==== ALTU            TEXT      0      0 05/23/1997      ON  OFF $SYS
*==== AP              TEXT      0      0 05/23/1997      ON  OFF $SYS
*==== ARCHIVE         TEXT      150     0 05/23/1997      ON  OFF $SYS
*==== ATT             TEXT      0      0 05/23/1997      ON  OFF $SYS
*==== ATTACH          TEXT      53      0 05/23/1997      ON  OFF $SYS
*==== ATTACHD         TEXT      40      0 05/23/1997      ON  OFF $SYS
```

Examples of SCREEN AUD

Display text:

```
=> screen aud,off
```

Display audit trail:

```
=> scr aud,on
```

Session after SCREEN AUD

```
=>
EDIT  ED51A.BIZUSRF                      SESS=A 1( 1) LINE=      0( 30564)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
***** -- TOP OF AUDIT --
***** .EDIT  DATE=06/13/1996, TIME=13:00:04, USER=MWD , TERM=LL1A
***** .INS   LOC= 9800, EXT=      5
***** |-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
***** | CASE          | specifies whether upper/lower case should be considered
***** |               | determining matches. If specified as "U", case is ignored
***** |               | "The" matches "the". If specified as "M", both case and le
***** |               | must match. If case is not specified, "U" is assumed.
***** .INS   LOC= 9800, EXT=      5
***** .INS   LOC= 8865, EXT=      4
***** The user must have edit access level for the MEM library. MEM ca
***** currently undergoing editing. MEM cannot be checkout slave assigned to
***** user. MEM cannot be a checkout master (have CHECK=ON) -- use
***** the checkout command to create a slave (working) copy.
```

Example of SCREEN BEF

Set before mode on, with six lines above the scale line:

```
=> screen bef,,6
```

Session after SCREEN BEF

```
=>
EDIT  ED51A.BIZUSRF                      SESS=A 1( 1) LINE= 2135( 17023)

***** which tend to be useful only to intensive users. The tables are se
***** into useful categories.
***** More information about each command may be found in the pages which
***** alphabetically by the command name. The shortest abbreviation for the
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
***** is also shown.
***** |-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
***** |               | Commands Used to Manipulate Members
***** |-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
***** | ALTER        | ALT | Alter member attributes.
***** | ATTACH        | ATT | Set library assumed for member access.
***** | AUDITCL       | AUDCL | Delete member's audit trail.
```


Examples of SCREEN LCA

Set LCA mode on, with the LCA area on the side of the screen indicated by predefined variable SIBDFLCA:

```
=> screen lca,on
```

Set LCA mode off.

```
=> scr lca,off
```

Session after SCREEN LCA,OFF

```
=>
EDIT  ED51A.BIZUSRF                      SESS=A 1( 1) LINE= 2135( 17023)
----|----1----|----2----|----3----|----4----|----5----|----6----|----7----|----
is also shown.

-----
|                                     Commands Used to Manipulate Members                                     |
|-----|
| ALTER | ALT | Alter member attributes. |
| ATTACH | ATT | Set library assumed for member access. |
| AUDITCL | AUDCL | Delete member's audit trail. |
| AUDITI | AUDI | Write specified line to audit trail. |
| AUDITRL | AUDRL | Back out member edits using audit trail. |
| AUDITSM | AUDSM | Summarize audit trail contents. |
|-----|
```

Toggle LCA mode, with the LCA area on the right side of the screen:

```
=> screen lca,,right
```

Session after SCREEN LCA,,RIGHT

```
=>
EDIT  ED51A.BIZUSRF                      SESS=A 1( 1) LINE= 2135( 17023)
----|----1----|----2----|----3----|----4----|----5----|----6----|----7----|
is also shown.

-----
|                                     Commands Used to Manipulate Members                                     |
|-----|
| ALTER | ALT | Alter member attributes. |
| ATTACH | ATT | Set library assumed for member access. |
| AUDITCL | AUDCL | Delete member's audit trail. |
| AUDITI | AUDI | Write specified line to audit trail. |
| AUDITRL | AUDRL | Back out member edits using audit trail. |
| AUDITSM | AUDSM | Summarize audit trail contents. |
|-----|
```

Examples of SCREEN SPLIT

Activate split-screen mode:

```
=> screen split,on
```

Deactivate split-screen mode:

```
=> scr sp,off
```

Switch split-screen mode from on to off, or from off to on:

```
=> scr
```

Activate split-screen mode. Logical screen #2 is to begin at row 15:

```
=> screen split,on,15
```

Session after SCREEN SPLIT,ON,15

```
=>
EDIT  ED51A.BIZUSRF                      SESS=A 1( 1) LINE= 2135( 17137)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* is also shown.
*====*
*====* -----
*====* |                                     Commands Used to Manipulate Members
*====* |-----
*====* | ALTER      | ALT   | Alter member attributes.
*====* | ATTACH      | ATT   | Set library assumed for member access.
*====* | AUDITCL      | AUDCL | Delete member's audit trail.
*====* | AUDITI      | AUDI  | Write specified line to audit trail.
*====* | AUDITRL      | AUDRL | Back out member edits using audit trail.
*====* | AUDITSM      | AUDSM | Summarize audit trail contents.
*====* |-----
~~~~~
EDIT  ED51A.BIZUSRF                      SESS=A 1( 1) LINE=    0( 17137)
< --|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF MEMBER --
*====* BIM-EDIT User Reference Manual
*====*
```

See Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual for more discussion.

SEARCH

Use SEARCH to position the current session at the "top of member" and then locate the first line having a specified string of characters in a specified column range.

SEARCH may also be entered as SRCH.

Optional Operands	
STR	is the string to search for. Up to 72 characters can be specified. If the value of STR begins with the backslash character (\), the search will use Extended Search Pattern rules, as described below. If STR is not specified, the value for STR and any other operands not specified will be inherited from previous search commands, also as described below.
ZONE	is the column range for the search. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to limit the search to columns 10 to 20, specify ZONE as "10-20". ZONE=5 is the same as ZONE=5-5, ZONE=5-* is the same as ZONE=5-253 and ZONE=-5 is the same as ZONE=1-5. If ZONE and STR are not specified, the ZONE specification is inherited from previous search commands, as described below. If ZONE, after any inheriting, is not specified, the current session zone is searched (see the ZONE operand of the DEFINE, ALTER, and SESS commands.)
CASE	specifies whether upper/lower case should be considered when determining matches. If specified as "U", case is ignored ("the" matches "The"). If specified as "M", both case and letters must match. If CASE and STR are not specified, the CASE specification is inherited from previous search commands, as described below. If CASE, after any inheriting, is not specified, "U" is assumed.

The search begins with the first line of the current session and proceeds until a match is found. An error message will display if no match is found.

See also the LOCATE command.

After SEARCH, the CURSOR command will place the cursor at the match.

SEARCH can be used with all session types. For example, LISTP sessions can be searched (for compilation error messages or other tell-tale results) and LIBRARY displays can be searched for words in member titles.

If your terminal is running under control of SNA, you may interrupt a long SEARCH command by pressing the ATTN key.

If the value of STR begins with a backslash (\), BIM-EDIT treats STR as an Extended Search Pattern, with certain characters having special meanings:

? matches any single character

- * matches any sequence of zero or more characters
- @ matches zero or more instances of the character that follows the @
- ! matches the beginning of the zone
- < matches the beginning of the zone or any non-alphanumeric character
- > matches the end of the zone or any non-alphanumeric character
- | separates two patterns, EITHER of which may match for a successful search
- + separates two patterns, BOTH of which must match for a successful search
- ~ reverses line-by-line success/failure of the pattern following it
- % marks the position on the line where the match is considered to occur
- \ suppresses the special meaning of the character that follows the \

See Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual for more information about Extended Search Patterns.

If STR is not specified on the command, BIM-EDIT uses as default operand values those from previous search commands. Using this feature, you can easily request searches which are similar or identical to the last one. To achieve this, BIM-EDIT stores a previous value for the STR, COL, CASE, and ZONE operands of the search commands (EXAMINE, FIND, FINDUP, LOCATE, LOCATEU, NFIND, NFINDUP, QUALIFY, SCAN or SEARCH). These previous values will be inherited by operands omitted from a search command which does not specify STR. Whenever a search command specifies STR, the previous values are set to those specified on that command or "not specified".

Use in a Procedure

Return Codes:

- | | |
|----|----------------------------------|
| OK | Successful. |
| AT | Command interrupted by ATTN key. |
| NF | String not found. |

SEARCH sets the predefined variable SSDCOL1 to the column number where the match is found (if in split-screen mode, and if the lower logical screen is primary, SSDCOL2 is set instead).

SEPARATE

Use SEPARATE to break apart current session text lines starting with the current line such that a paragraph in a specified column range becomes separate lines with a phrase on each line.

SEPARATE may also be entered as SEP.

Optional Operands	
ZONE	is the column range in which to separate text. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to separate text within columns 14 to 77, specify ZONE as "14-77". ZONE=5 is the same as ZONE=5-5, ZONE=5-* is the same as ZONE=5-253 and ZONE=-5 is the same as ZONE=1-5. If ZONE is not specified, text is separated in the current session zone (see the ZONE operand of the DEFINE, ALTER, and SESS commands.)

SEPARATE updates a group of lines composing a paragraph such that each phrase begins on a new line; a new line is begun after most punctuation marks. In addition, SEPARATE will limit the amount of text on any line such that at least 20 blanks are provided at the end of each line, thus facilitating character insertion.

Separation begins with the current line and proceeds until an end-of-paragraph indication. End-of-paragraph is indicated by a line with a blank, a period (.), an asterisk (*), a colon (:), or a dash (-) in the first position of the zone. The current line is always separated, even if it has an end-of-paragraph indication.

Normally, SEPARATE will ignore extraneous blanks (more than one blank) when processing text. However, SEPARATE will retain blanks that serve to indent the first line of a paragraph.

SEPARATE updates text only within the ZONE. Since SEPARATE leaves text outside the ZONE intact, it can often be used on text in tables or lists. For example, when a BIM-EDIT operand table description is updated, the following command is used to separate the text:

```
=> separate 14-77
```

The operand name and the vertical bar characters remain unaffected since they are outside the ZONE.

SEPARATE is often used in conjunction with the FORMAT command. SEPARATE is used to update a group of lines composing a paragraph such that each phrase begins on a new line. Editing a paragraph is much easier after the SEPARATE processing. When editing is complete, the paragraph is formatted via FORMAT.

Use in a Procedure

SEPARATE normally returns OK. (See SIBRETCD.)

Example

The following example assumes the current session ZONE is 1-79.

EDIT session before SEPARATE command

```
=> separate
EDIT 3217.WORK                                SESS=A 1( 1) LINE= 4442( 11247)
----|----1----|----2----|----3----|----4----|----5----|----6----|----7----|----
FORMAT is often used in conjunction with the SEPARATE command. SEPARATE is
used to update a group of lines composing a paragraph such that each phrase
begins on a new line. Editing a paragraph is much easier after the SEPARATE
processing. When editing is complete, the paragraph is formatted via FORMAT.

Use in a Procedure
-----

SEPARATE always returns OK.
```

EDIT session after SEPARATE command

```
=>
EDIT 3217.WORK                                SESS=A 1( 1) LINE= 4442( 11251)
----|----1----|----2----|----3----|----4----|----5----|----6----|----7----|----
FORMAT is often used in conjunction with the SEPARATE
command.
SEPARATE is used to update a group of lines composing a
paragraph such that each phrase begins on a new line.
Editing a paragraph is much easier after the SEPARATE
processing.
When editing is complete,
the paragraph is formatted via FORMAT.

Use in a Procedure
-----

SEPARATE always returns OK.
```

SEQCHECK

Use SEQCHECK to check the order of current session lines according to the characters in specified columns for a specified number of lines starting with the current line.

SEQCHECK may also be entered as SEQCK.

Required Operands	
KEY	<p>specifies the sequence key. Sequencing can be based upon up to six column ranges, so long as the combined key length does not exceed 253. If multiple column ranges are specified, column ranges are separated from each other with a comma and the entire KEY is enclosed within apostrophes. Each column range specification is composed of up to three parts, separated by dashes:</p> <ul style="list-style-type: none"> the base column number (from 1 through 253). the limit column number (from 1 through 253). the sort order (A=ascending, D=descending). <p>The base and limit column numbers are always specified. The sort order need be specified only if descending order is desired</p> <p>The following are all valid sequence keys:</p> <pre>1-12 1-12-A 1-12-D '1-12, 40-42 ' '1-12-D, 40-42-A '</pre>
FCT	<p>specifies the number of lines to sequence check. If FCT is specified as an asterisk (*), all of the lines through the end of the session will be checked.</p>
IGN	<p>can be specified to cause certain lines to be ignored in the sequence checking process if they contain, in column 1, the character specified for this operand.</p>

Order is determined by the EBCDIC collating sequence. Most special characters precede all lower case letters, which precede all upper case letters, which precede all digits.

If your terminal is running under control of SNA, you may interrupt this command by pressing the ATTN key.

Use in a Procedure

Return Codes:

OK	Successful.
NF	String not found.
SQ	Sequence error detected. The current session line will be set to the last line still in sequence and a message will be displayed.
AT	Interrupted by ATTN key.

SV Invalid operand specification.

Example

Sequence check all lines on columns 1 thru 8:

```
=> seqck 1-8
```

Sequence check 10 lines on columns 73 thru 80, ignoring any lines that begin with an '*':

```
=> seqck 73-80 fct=10 ign=*
```


SESS

Use SESS to alter session attributes.

SESS may also be entered as SS.

Optional Operands	
CASE	is the alphabetic case of the session. Specify "U" if updated text lines are to be translated to upper case. Specify "M" if translation is not to occur. Only new and updated lines are affected by this setting.
NULLS	specifies whether trailing blanks on a line are to be replaced with nulls when displaying the session. "ON" specifies that trailing blanks will be replaced, "OFF" specifies that they will not, and "TOG" reverses the current setting. When trailing blanks are replaced with nulls, the 3270 insert key can be used. However, if text is entered to the right of null character(s) on a line, the text ends up shifted left since the nulls are not transmitted to BIM-EDIT.
SEQ	is a column range where the RESEQ command places sequence numbers. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to place sequence numbers in columns 73 through 80, specify SEQ as "73-80". The SEQ range cannot be wider than 8 columns.
TCOL	is a list of columns used for tabbing. Up to 12 tab settings may be specified. They are entered in the format "cc-cc-cc". Each setting consists of one to three digits which specify the column where the text will be placed. Tabbing range is from 1 to 253. Tab columns must be entered in ascending order. For a more extensive discussion of tabbing, see Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual.
ZONE	is a column range used by the edit and search commands (such as CHANGE, CENTER, EXAMINE, FORMAT, LOCATE, QUALIFY, etc.) if not specified explicitly on the command. For example, the CHANGE command will change occurrences of one string to another within the current ZONE only. Enter ZONE in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to set the zone to columns 1 through 72, specify ZONE as "1-72". ZONE=5 is the same as ZONE=5-5, ZONE=5-* is the same as ZONE=5-253 and ZONE=-5 is the same as ZONE=1-5.
FCOL	is a column used by the FIND, FINDUP, NFIND, and NFINDUP commands. For example, the FIND command scans forward for a line with a specified string starting in the FCOL column.

The SESS command sets attributes only for the current session. Initially, the attributes of a session are set from the attributes of the member. When the session is ended, session settings are lost. To permanently change attributes for a member, use the ALTER command.

Use in a Procedure

SESS normally returns OK. (See SIBRETCD.)

Session attributes can be accessed within a procedure using the SSD predefined variables.

Examples

Allow mixed case entry:

```
=> sess case=m
```

Set tabs for a COBOL program:

```
=> sess tcol=8-12-16-20-32-40
```

SET

Use SET to set a PF key command or system control value.

Required Operands	
DEST	<p>is one of the following:</p> <p>PF1-24 specifies the command for a PF key is to be set.</p> <p>PA1-3 specifies the command for a PA key is to be set.</p> <p>CLEAR specifies the command for the CLEAR key is to be set.</p> <p>ENTER specifies the command for the ENTER key is to be set. This command will be used if ENTER is hit when no updates or commands have been entered and there isn't a message displayed on the information line.</p> <p>SIBCCTL specifies "extended cursor control" is to be set.</p> <p>SIBDFLCA specifies the "default LCA side" is to be set.</p> <p>SIBHEXC specifies the "hex escape character" is to be set.</p> <p>SIBTABC specifies the "logical tab character" is to be set</p>
SRC	<p>is the data value to assign. Its use depends on DEST:</p> <p>PF1-24, PA1-3, CLEAR, or ENTER SRC is the text of commands to be executed when the specified key is hit. The value of SRC would normally have to be enclosed in quotes in this case.</p> <p>SIBCCTL SRC is the cursor control option. If it is "1", the cursor will be positioned to the first added or inserted line. If it is "2", cursor control is extended so that modified lines and deleted lines cause cursor positioning.</p> <p>SIBHEXC SRC is the character to be used for hex escape. For more detail, see Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual.</p> <p>SIBDFLCA SRC is either "R" if you want the LCA on the right side when you start a session or "L" if you want the LCA on the left.</p> <p>SIBTABC SRC is the character you want to use to represent a tab character in a line. For more detail, see Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual.</p>

Optional Operands	
OPT	<p>specifies a PF key option. OPT may be specified as one of the following:</p> <p>AFTER PF key data will be processed after the command line.</p> <p>BEFORE PF key data will be processed before the command line.</p> <p>IGNORE PF key data will not be processed if there is data on the command line.</p> <p>ONLY Only PF key data will be processed. Data on the command line is used if the 'data insert' character is found. If the 'data insert' character is not found, the data on the command line is ignored.</p> <p>If omitted, AFTER is assumed.</p>

The KEYS command may be easier to use than the SET command to set the commands for function keys. KEYS displays the current settings and allows you to change the settings by overtyping them.

It is possible to have command line input data (up to four separate parameters) merged with PF key data before the command string is presented to BIM-EDIT. The parameter data insertion will take place when a PF key is depressed and the associated PF key data has a reverse quote (') imbedded within it. Any number of reverse quote marks can appear with the PF key definition. As each reverse quote mark is encountered in the PF key data it is replaced with a parameter from the command line input. If only one parameter is entered on the command line all reverse quote marks encountered in the PF key data will be replaced with that single parameter. If two command line parameters are entered and there are five occurrences of the reverse quote mark in the PF key data, parameter one from the command line will replace the first reverse quote mark and command line parameter two will replace reverse quote two through five. The processing option for the data insertion must be set to "ONLY".

(This description omits SET features which are used when writing procedures. SET is discussed completely in Chapter 3, Advanced User Commands, in the BIM-EDIT System Reference Manual.)

Use in a Procedure

SET normally returns OK. (See SIBRETC.D.)

Examples

Set the HEX character:

```
=> set sibhexc,@
```

Set the PF12 function key:

```
=> set pf12,'rot -',opt=only
```

Clear PF21:

```
=> set pf21, ''
```

SHIFT

Use SHIFT to shift current session text either right or left a specified number of columns in a specified column range for a specified number of lines starting with the current line.

SHIFT may also be entered as SH.

Required Operands	
COL	number preceded or followed by a plus(+) or minus(-) sign. The plus sign causes the text to shift to the right, the minus sign causes the text to shift to the left. The amount shifted is the numeric value entered for COL. If only the + or - is entered, the text will be shifted by one column.

Optional Operands	
FCT	is the number of lines for which text shifting will occur. If FCT is specified as *, all lines are shifted through the end of the session. If FCT is not specified, only the current line will be shifted.
ZONE	is the column range in which to shift text. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to limit the shift to columns 11 to 66, specify ZONE as "11-66". ZONE=5 is the same as ZONE=5-5, ZONE=5-* is the same as ZONE=5-253 and ZONE=-5 is the same as ZONE=1-5. If ZONE is not specified, text is shifted in the current session zone (see the ZONE operand of the DEFINE, ALTER, and SESS commands.)

Any text shifted out of the zone is lost. That is, SHIFT can be used to eliminate text in a column range. To copy text in a column range, use PROPAGATE.

SHIFT is the command line equivalent of the LCA < and > commands.

Use in a Procedure

SHIFT normally returns OK. (See SIBRETCDD.)

Example

EDIT session before SHIFT command

```
=> shift -6,2,9-20
EDIT 1169.CFACT                                SESS=A 1( 1) LINE= 17( 234)
----|----1----|----2----|----3----|----4----|----5----|----6----|----7--
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
```

EDIT session after SHIFT command

```
=>
EDIT 1169.CFACT                                SESS=A 1( 1) LINE=   17(   234)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==* SEE THE BROWN          FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE BROWN          FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
```

SHOW

Use SHOW to display BIM-EDIT general information or PF key settings. When SHOW is invoked online, it creates a DISP session for the results.

Optional Operands	
OBJ	<p>Specifies what display to create. Some options are:</p> <p>GEN BIM-EDIT frequently referenced general information. Some of the fields displayed are: user id, terminal id, "last referenced member", "last referenced POWER job entry", or "last referenced JES data sets".</p> <p>PF, PFK, KEY, or KEYS</p> <p> the functions and processing options assigned to the PF keys and the special keys CLEAR, ENTER PA1, PA2, and PA3.</p> <p>If OBJ is not specified, "GEN" is assumed.</p>

(The KEYS command may be more convenient to use than the SHOW command to display the commands for function keys. The KEYS command displays the current settings and allows you to change the settings by overtyping them.)

(This description omits SHOW features which are used when writing procedures. SHOW is discussed completely in Chapter 3, Advanced User Commands, in the BIM-EDIT System Reference Manual.)

Use in a Procedure

SHOW normally returns OK. (See SIBRETC D.)

If SHOW is invoked in batch utility, the display destination is controlled by predefined variable SIBOUTPT.

Examples

Show keys:

```
=> show keys
```

Show general information:

```
=> show
```


SORT

Use SORT to rearrange the order of current session lines according to the characters in specified columns for a specified number of lines starting with the current line.

Required Operands	
KEY	<p>specifies the sort key. Sorting can be based upon up to six column ranges, so long as the combined key length does not exceed 253. If multiple column ranges are specified, column ranges are separated from each other with a comma and the entire KEY is enclosed within apostrophes. Each column range specification is composed of up to three parts, separated by dashes:</p> <ul style="list-style-type: none"> the base column number (from 1 through 253). the limit column number (from 1 through 253). the sort order (A=ascending, D=descending). <p>The base and limit column numbers are always specified. The sort order need be specified only if descending order is desired</p> <p>The following are all valid sort keys:</p> <pre>1-12 1-12-A 1-12-D '1-12,40-42' '1-12-D,40-42-A'</pre>
FCT	<p>specifies the number of lines to sort. If FCT is specified as an asterisk (*), all of the lines through the end of the session will be sorted.</p>

Order is determined by the EBCDIC collating sequence. Most special characters precede all lower case letters, which precede all upper case letters, which precede all digits. To obtain a case-insensitive SORT of mixed case data, use PROPAGAT to copy the key data to an unused column range and then use UPPERCAS or LOWERCAS to translate the letters.

If your terminal is running under control of SNA, you may interrupt a long SORT command by pressing the ATTN key. If the SORT is aborted in this manner, all updates made to the member since the start of the edit session will be lost.

Use in a Procedure

SORT normally returns OK. (See SIBRETC D.)

Example

EDIT session before SORT command

```
=> sort 1-14 *
EDIT  RND.COMMSTDS                      SESS=A 1( 1) LINE=    0(   19)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==== -- TOP OF MEMBER --
*==== ISO 8802-2      Logical Link Control
*==== ISO 9314-2      FDDI Token Ring Media Access Control
*==== ANSI X3.167     LDDI Data Link Protocol and Network Specific Sub-Layer
*==== ISO 8802-3      CSMA/CD Media Access Control
*==== ISO 8802-4      Token Bus Media Access Control
*==== IEEE 802.1d     MAC Bridges
*==== ISO 8802-5      Token Ring Media Access Control
*==== ISO 7776        Multilink Procedure (MLP)
*==== CCITT X.25       Multilink Procedure (MLP)
*==== ISO 8802-7      Slotted Ring Access Method
*==== ISO 7776        Single Link Procedure LAP/LAPB
```

EDIT session after SORT command

```
=>
EDIT  RND.COMMSTDS                      SESS=A 1( 1) LINE=    0(   19)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==== -- TOP OF MEMBER --
*==== ANSI X3.167     LDDI Data Link Protocol and Network Specific Sub-Layer
*==== CCITT Q.703      Signalling System No. 7/Signalling Link Protocol
*==== CCITT T.71       Half Duplex LAPB Extension (HDTM)
*==== CCITT X.25       Multilink Procedure (MLP)
*==== CCITT X.25       Single Link Procedure LAP/LAPB
*==== IEEE 802.1d     MAC Bridges
*==== ISO 3309         HDLC Frame Structure
*==== ISO 4335         HDLC Elements of Procedures
*==== ISO 7776        Multilink Procedure (MLP)
*==== ISO 7776        Single Link Procedure LAP/LAPB
```

SPLIT

Use SPLIT to split the current line into two lines at a specified column or string.

SPLIT may also be entered as SP.

(Note: to enter split-screen mode, use the SCREEN command SPLIT option.)

The cursor can be used to designate the location where the SPLIT is to occur or optionally the following parameters can be used. If the cursor is used the text starting at the column where the cursor is positioned will become the start of the next line.

Required Operands	
LOC	<p>is the split point, and may be a number or a character string:</p> <ul style="list-style-type: none"> If LOC is a number from 2 to 253, the text occurring at that column will become the start of the second line. If LOC is a character string, the first occurrence of the string will become the start of the second line. The character string can be up to 36 characters long but cannot start in column 1 of the text line. <p>If LOC is not specified and the cursor does not fall with the text area on the screen an error message will be displayed.</p>
CASE	<p>specifies whether upper/lower case should be considered when determining matches. If specified as "U", case is ignored ("the" matches "The"). If specified as "M", both case and letters must match. If CASE is not specified, "U" is assumed.</p>

All text up to the split point is retained as the current line, and all text after the split point will become the second line.

Use in a Procedure

Return Codes:

OK Successful.
NF String not found.

Examples

Note: In the following examples the underscore "_" designates the cursor.

EDIT session before the SPLIT command

```
=> split
EDIT 1283.CPACT                      SESS=A 1( 1) LINE= 17( 19)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
-- END OF MEMBER --
```

EDIT session after the SPLIT command

```
=>
EDIT 1283.CPACT                                SESS=A 1( 1) LINE= 17( 20)
----|----1----|----2----|----3----|----4----|----5----|----6----|----7--
*==== SEE THE QUICK BROWN FOX
*==== JUMP OVER THE LAZY BROWN DOG
*==== SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==== SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
-- END OF MEMBER --
```

EDIT session before the SPLIT command

```
=> split 25
EDIT 1283.CPACT                                SESS=A 1( 1) LINE= 17( 19)
----|----1----|----2----|----3----|----4----|----5----|----6----|----7--
*==== SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==== SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==== SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
-- END OF MEMBER --
```

EDIT session after the SPLIT command

```
=>
EDIT 1283.CPACT                                SESS=A 1( 1) LINE= 17( 20)
----|----1----|----2----|----3----|----4----|----5----|----6----|----7--
*==== SEE THE QUICK BROWN FOX
*==== JUMP OVER THE LAZY BROWN DOG
*==== SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==== SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
-- END OF MEMBER --
```

EDIT session before the SPLIT command (LOC is a string)

```
=> split 'the lazy'
EDIT 1283.CPACT                                SESS=A 1( 1) LINE= 17( 19)
----|----1----|----2----|----3----|----4----|----5----|----6----|----7--
*==== SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==== SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==== SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
-- END OF MEMBER --
```

EDIT session after the SPLIT command

```
=>
EDIT 1283.CPACT                                SESS=A 1( 1) LINE= 17( 20)
----|----1----|----2----|----3----|----4----|----5----|----6----|----7--
*==== SEE THE QUICK BROWN FOX JUMP OVER
*==== THE LAZY BROWN DOG
*==== SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==== SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
-- END OF MEMBER --
```

SQUEEZE

Use SQUEEZE to remove embedded blanks in a specified column range for a specified group of lines.

SQUEEZE may also be entered as SQUEZ or SQU.

Optional Operands	
ZONE	is the column range in which embedded blanks are to be removed. Enter it in the format xxx-yyy where xxx and yyy are numbers between 1 and the session maximum line width. For example, ZONE=11-20 signifies columns 11 to 20. ZONE=5 is the same as ZONE=5-5, ZONE=5-* specifies columns 5 through the session maximum line width, and ZONE=-5 is the same as ZONE=1-5. ZONE width must be at least 2. Default: current session zone.
FCT	specifies the number of lines to process. If FCT is specified as an asterisk (*), all of the lines through the end of the session will be processed. If FCT is not specified, only the current line will be processed.
LEAVE	Is the number of blanks to be left between non-blank characters. Default: 0.

Use in a Procedure

SQUEEZE normally returns OK. (See SIBRETC D.)

Example

EDIT session before SQUEEZE command

```
=> squeeze 10-25,5,19
EDIT DOC.C                                SESS=A 1( 1) LINE=      8(   93)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*      AC N  = PMA AC;                                COMMENT LINE
*====*      NAME = PMA NAME;
*====*      ADDR1 = PMA ADDR1;
*====*      ADDR2 = PMA ADDR2;
*====*      CITY = PMA CITY;
```

EDIT session after SQUEEZE command

```
=>
EDIT DOC.C                                SESS=A 1( 1) LINE=      8(   93)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*      AC N  = PMA AC                                COMMENT LINE
*====*      NAME = PMA NAME
*====*      ADDR1 = PMA ADDR1;
*====*      ADDR2 = PMA ADDR2;
*====*      CITY = PMA CITY;
```

STACK

Use STACK to copy a specified number of lines from the current session into \$STACK starting with the current line.

STACK may also be entered as ST.

Optional Operands	
FCT	specifies the number of lines to copy. If FCT is not specified, no lines are copied. If FCT is specified as asterisk (*), the lines through the end of the session are copied.
OPT	If OPT is not specified, or if OPT is specified as CLEAR or CL, \$STACK is first cleared. If APPEND or AP is specified, \$STACK is not cleared.

STACK, in combination with GET \$STACK or the LCA I or B commands, is generally used for moving or copying lines. It can also be used with the MERGE command or the LCA G or J commands to overlay member lines. Further, the \$STACK area from the STACK command can be the object of EXECUTE, MAIL, PRINT, or SUBMIT.

STACK is the command line equivalent of the LCA C and K commands. The effect of an LCA M or N command could be achieved by a STACK command followed by a DELETE command.

Use in a Procedure

STACK normally returns OK. (See SIBRETC D.)

Examples

After clearing \$STACK, copy 20 lines to it:

```
=> stack 20
```

Append 740 lines from the current session to \$STACK:

```
=> st 740,ap
```

Clear \$STACK:

```
=> stack ,cl
```

STACKI

Use STACKI to write a specified line of text (sometimes called the "immediate" line) to \$STACK.

Required Operands	
LINE	is the line to write to \$STACK.

STACKI appends the line to \$STACK. It does not clear \$STACK.

Use in a Procedure

STACKI normally returns OK. (See SIBRETC D.)

Example

Write a single line to \$STACK:

```
=> stacki 'sk=0010,lp=03'
```

STAMPCL

Use STAMPCL to clear a member's date stamp information.

Required Operands	
MEM	is the member for which the stamp information is to be cleared.
DATE	specifies how much stamp information to clear. If DATE is specified as an asterisk (*), all stamps are cleared. Otherwise, DATE is entered either in the format "mm/dd/yy", "mm/dd/yyyy", "dd/mm/yy" or "dd/mm/yyyy" depending upon the date format set by the System Administrator. In this case, only those stamps with a date up to and including DATE will be cleared. Those stamps with dates greater than DATE will not be cleared.

For more information about member stamping, see Chapter 6, Advanced Techniques, in the BIM-EDIT User Reference Manual.

STAMPCL sets the "last referenced member".

When Valid

If your System Administrator has set MMPSPCTL to a value of "1", this command is only available to the System Administrator. Otherwise, the user must have DEF access level for the MEM library. MEM cannot be currently undergoing editing. MEM cannot be part of a checkout relationship.

Use in a Procedure

Return Codes:

OK	Successful.
CK	Member is part of a checkout relationship.
ED	Member is being edited.
NF	Member not found.
SC	Inadequate access level.

STAMPCL sets the TXM variables to the attributes of MEM.

Examples

Clear the stamps for member APC0100:

```
=> stampcl apc0100,*
```

Clear all stamps for member XIR2500 with dates up to 12/31/96:

```
=> stampcl xir2500,12/31/1996
```


SUBMIT (VSE version)

Use SUBMIT to submit a member for batch processing.

SUBMIT may also be entered as SUB.

Optional Operands	
MEM	is the member to be submitted for processing. If MEM is not specified, the "last referenced member" is used. MEM can also be "\$STACK" to submit the contents of the stack area, or "/" to submit the current session.
CLASS	is the job class.
DISP	is the job disposition.
USER	is the job user information.
LSTCLASS	is the class assigned to the list output.
LSTDISP	is the disposition assigned to the list output.
PUNCLASS	is the class assigned to the punch output.
PUNDISP	is the disposition assigned to the punch output.
NTFY	<p>is the job completion notification option. If omitted, the default notify option from the user definition is used.</p> <p>YES send completion message to the user submitting the job.</p> <p>NO don't send any completion message (default).</p> <p>xxxxxxx send completion message to BIM-EDIT user xxxxxxxx.</p>
PSWD	If the member is password protected, the password must be entered. The password may be up to 8 characters in length consisting of any characters.

SUBMIT is often invoked through the PROCESS command, which in turn is invoked from either the command line or the LCA S command. See PROCESS for more information.

Submitting the current session is useful to process temporary modifications to standard members. Simply END NOSAVE after the SUBMIT.

If your terminal is running under control of SNA and you are running VSE/SP release 2.1 or later, you may abort a long SUBMIT command by pressing the ATTN key.

The NTFY operand is only valid if you are running VSE/ESA release 2.1 or above. The completion message is sent to the specified user by the BIM-EDIT mail facility.

The operands CLASS, DISP, USER, LSTCLASS, LSTDISP, PUNCLASS, and PUNDISP are relevant only where the member being submitted does not begin with a POWER

JOB line. Defaults for these operands are determined in the SUBMIT procedure, and are usually unique to a BIM-EDIT site.

SUBMIT sets the "last referenced member".

The above describes the functioning of SUBMIT as distributed with BIM-EDIT. SUBMIT is often modified for a particular site's needs. Check with your System Administrator regarding the operation of SUBMIT at your site.

When Valid

The user must have LIST access level for the MEM library. MEM must not be currently undergoing editing. Check with your System Administrator to determine any other restrictions on SUBMIT at your site.

Use in a Procedure

Return Codes:

OK	Successful.
AT	Command interrupted by ATTN key.
ED	Member is being edited.
NF	Member not found.
SC	Inadequate access level.

SUBMIT sets the TXM variables to the attributes of MEM.

SUBMIT is implemented as the system procedure BIPSUBM. SUBMIT handles the actual submission by invoking the SUBMITF command.

Example

Submit the member OMJBKPG:

```
=> submit omjbkpg
```

SUBMIT (MVS version)

Use SUBMIT to submit a member for batch processing.

SUBMIT may also be entered as SUB.

Optional Operands	
MEM	is the member to be submitted for processing. If MEM is not specified, the "last referenced member" is used. MEM can also be "\$STACK" to submit the contents of the stack area, or "/" to submit the current session.
PSWD	If the member is password protected, the password must be entered. The password may be up to 8 characters in length consisting of any characters.

SUBMIT is often invoked through the PROCESS command, which in turn is invoked from either the command line or the LCA S command. See PROCESS for more information.

Submitting the current session is useful to process temporary modifications to standard members. Simply END NOSAVE after the SUBMIT.

If your terminal is running under control of SNA, you may abort a long SUBMIT command by pressing the ATTN key.

SUBMIT sets the "last referenced member".

SUBMIT is often modified for a particular site's needs. Check with your System Administrator regarding the operation of SUBMIT at your site.

When Valid

The user must have LIST access level for the MEM library. MEM must not be currently undergoing editing. Check with your System Administrator to determine any other restrictions on SUBMIT at your site.

Use in a Procedure

Return Codes:

OK	Successful.
AT	Command interrupted by ATTN key.
ED	Member is being edited.
NF	Member not found.
SC	Inadequate access level.

SUBMIT sets the TXM variables to the attributes of MEM.

SUBMIT is implemented as the system procedure BIPSUBM. SUBMIT handles the actual submission by invoking the SUBMITF command.

Example

Submit the member OMJBKPG:

SUBMIT (MVS version)

```
=> submit omjbkpg
```

SUBMITD (MVS Only)

Use SUBMITD to submit a PDS member for batch processing.

SUBMITD may also be entered as SUBD.

Optional Operands	
MEMD	the name of the PDS member to be submitted. The currently attached PDS (see ATTACHD) will be used unless the PDS name is prefixed to the member name, that is, pdsname.member or pdsname(member). If MEMD is not specified, the "last referenced PDS member" will be submitted.

If your terminal is running under control of SNA, you may interrupt a long SUBMITD command by pressing the ATTN key.

SUBMITD sets the "last referenced PDS member".

SUBMITD may be modified for a particular site's needs. Check with your System Administrator regarding the operation of SUBMIT at your site.

When Valid

The user must have LIST access level for the MEMD library. Check with your System Administrator to determine any other restrictions on SUBMIT at your site.

Use in a Procedure

Return Codes:

OK	Successful.
AT	Command interrupted by ATTN key.
ED	Member is being edited.
NF	Member not found.
SC	Inadequate access level.

SUBMITD is implemented as the system procedure BIPSUBD. SUBMITD handles the actual submission by invoking the SUBMIT command.

Example

Submit member TEST2 in the current PDS:

```
=> submitd test2
```

Submit member JRNLRUN in PDS USERLIB3.GENL.C:

```
=> subd userlib3.genl.c.test2
```

Submit member PROG428 in PDS SYSLIB.PROBLEM:

```
=> subd syslib.problem(prog428)
```

SWAP

Use SWAP to position the current session to the alternate position marker. In SCREEN SPLIT mode, use SWAP to exchange the upper and lower logical screens.

SWAP may also be entered as SW.

SWAP has no operands.

Two position markers within a session are always maintained by BIM-EDIT. In split-screen mode (SCREEN SPLIT=ON) with both logical screens displaying the same session, both positions are visible. In single-screen mode, only one position is visible at a time.

In single-screen mode, the SWAP command will cause the positions to be exchanged. This has the effect of positioning at the alternate marker. SWAP allows you to easily switch back and forth between two different areas of a session. (You can use it to set "book marks".)

In split-screen mode, SWAP exchanges the upper and lower screens.

Use in a Procedure

SWAP normally returns OK. (See SIBRETC D.)

Example

Exchange position 1 with position 2:

```
=> swap
```

TOP

Use TOP to position the current session before the furthest backward (up-most) line in the session.

TOP may also be entered as T.

TOP has no operands.

Use in a Procedure

TOP normally returns OK. (See SIBRETCD.)

Example

Position at the top of the session:

```
=> top
```

UNDO

Use UNDO to back out edit sessions that have taken place against a member that has an audit trail.

The audit information associated with the UNDO change is also removed and replaced with a ".UNDO" audit statement that contains the date, time, userid and terminal used to make the change.

Optional Operands	
COUNT	specifies the number of edit sessions to be removed from the text and audit information. The default is 1.

To use UNDO you must first start an EDIT session. You can be on either an EDIT display of the member text or a display of the audit information (SCR AUD).

UNDO locates the audit information. It then looks for the specified number of .EDIT audit statements starting at the bottom of the member and searching backward.

Once the UNDO has been issued it is suggested that you verify that the status of the text is what you intended. If it is, SAVE the member. Otherwise, enter "END NOSAVE".

UNDO writes all the deleted audit lines to the \$LOG.

Use in a Procedure

UNDO always returns OK.

UP

Use UP to position the current session backward (toward the top) a specified number of lines.

UP may also be entered as U.

Optional Operands	
FCT	specifies the number of lines to position backward by. If FCT is not specified, BIM-EDIT will position backward one line.

It is not an error if FCT exceeds the number of lines to the top of the session.

Use in a Procedure

Return Codes:

OK	Successful
EF	Already positioned at the first line of the session (this only occurs when UP is executed from a procedure).

Examples

Position backward 1 line:

```
=> up
```

Position backward 65 lines:

```
=> u 65
```

Use UP to position the current session backward (toward the top) a specified number of lines.

UP may also be entered as U.

Optional Operands	
FCT	specifies the number of lines to position backward by. If FCT is not specified, BIM-EDIT will position backward one line.

It is not an error if FCT exceeds the number of lines to the top of the session.

Use in a Procedure

Return Codes:

OK	Successful
EF	Already positioned at the first line of the session (this only occurs when UP is executed from a procedure).

Examples

Position backward 1 line:

```
=> up
```

Position backward 65 lines:

=> u 65

UPPERCAS

Use UPPERCAS to translate current session text from lowercase letters to uppercase letters in a specified column range for a specified number of lines starting with the current line.

UPPERCAS may also be entered as UPPER or UPP.

Optional Operands	
FCT	specifies the number of lines for which text is to be translated. If FCT is specified as an asterisk (*), text will be translated through the end of the session. If FCT is not specified, text will be translated for the current line only.
ZONE	is the column range in which to translate text. Enter it in the format "xxx-yyy" where xxx and yyy are numbers between 1 and 253 separated by a dash (-). For example, to translate text within columns 11 to 66, specify ZONE as "11-66". ZONE=5 is the same as ZONE=5-5, ZONE=5-* is the same as ZONE=5-253 and ZONE=-5 is the same as ZONE=1-5. If ZONE is not specified, text is translated in the current session zone (see the ZONE operand of the DEFINE, ALTER, and SESS commands.)

The LOWERCAS command can be used to translate from uppercase to lowercase.

UPPERCAS is the command line equivalent of the LCA U command.

Use in a Procedure

UPPERCAS normally returns OK. (See SIBRETC D.)

Examples

Translate text to uppercase through the end of the session:

```
=> uppercas *
```

Translate 10 lines to uppercase:

```
=> upp 10
```

VIEW

Use VIEW to position the current session display left or right.

Optional Operands	
COL	<p>The value for COL may be entered one of three ways:</p> <p>xxx An unsigned number from 1 to 253. The number will become the left-most column displayed for the session.</p> <p>+xxx A number from 1 to 253 preceded by a plus sign(+). The current left margin will be increased by the number given, up to a maximum left margin of 253. The result is a shift of the viewing range "xxx" columns to the right.</p> <p>-xxx A number from 1 to 253 preceded by a minus sign(-). The current left margin will be decreased by the number given, to a minimum left margin of 1. The result is a shift of the viewing range "xxx" columns to the left.</p> <p>If COL is not specified, a value of "1" is assumed.</p>

The scale line on the screen displays the current viewing range.

VIEW is especially useful with LISTP displays. You may find it desirable to set PF keys to "VIEW 1" and "VIEW 53" to allow quickly changing your view.

Use in a Procedure

VIEW normally returns OK. (See SIBRETC.D.)

Examples

Screen before VIEW command ("xxx" format)

```
=> view 11
EDIT  1178.CFACT                                SESS=A 1( 1) LINE=   17(   234)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
```

Screen after VIEW command ("xxx" format)

```
=>
EDIT  1178.CFACT                                SESS=A 1( 1) LINE=   17(   234)
-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----|-----8--
*==* ICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* ICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* ICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* ICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* ICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* ICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
```

Screen before VIEW command ("-xxx" format)

```
=> view -10
EDIT 1178.CFACT                                SESS=A 1( 1) LINE= 17( 234)
-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----|-----8--
*==* ICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* ICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* ICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* ICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* ICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* ICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
```

Screen after VIEW command ("-xxx" format)

```
=>
EDIT 1178.CFACT                                SESS=A 1( 1) LINE= 17( 234)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
```

VTOC (VSE Only)

Use VTOC to create a display session of the contents of the VTOC for a DASD volume.

Optional Operands	
VOL	is the VOLID to be displayed. If omitted, a list is generated of all VOLIDs defined to VSE.

If a list of VOLIDs is generated as a result of not specifying the VOL operand, you can display the VTOC of a specific VOLID by using the LCA L command.

The files are listed in starting track/block sequence.

The following fields are displayed for each entry in the VTOC:

FILE ID	44 character file name
XT/SQ	EXTENT sequence number for multi-extent datasets
NBR. OF TRACKS	For CKD or ECKD files, this is the number of tracks allocated.
NBR. OF BLOCKS	For FBA, this is the number of FBA blocks allocated.
FILE TYPE	Type of dataset defined for the extent: SYST System dataset VSAM VSAM dataset/space/catalog SAM Sequential Access Method dataset DAM Direct Access Method dataset UNDF Unknown dataset type. (This field is preceeded by an '*' if the file is expired.)
STARTING CCCC.HH TRACK	For CKD and ECKD, this specifies where the EXTENT begins, as an actual CCCC.HH and a relative track number.
STARTING BLOCK	For FBA, this specifies where the EXTENT begins, as a relative FBA block number.
ENDING CCCC.HH TRACK	For CKD and ECKD, this specifies where the EXTENT ends, as an actual CCCC.HH and a relative track number.
ENDING BLOCK	For FBA, this specifies where the EXTENT ends, as a relative FBA block number.
CREATE DATE	Julian date, in CCYY/DDD format of when the dataset was first defined
EXPIRE DATE	Julian date of when the dataset is set to expire. (If it is already expired, the word 'EXPIRED' will appear following this date, and an '*' will preceed the FILE TYPE field.)

Not all fields will display on an 80-column screen. You will either need a Model-5 display, or will need to use the VIEW command to shift some fields into view.

*** VOL1 AND IPL *** This line defines the space reserved for the VOL1 record.

*** VTOC EXTENT *** This line defines the space reserved for the VTOC.

*** FREE SPACE *** One or more of these lines will appear to indicate where free space appears on the volume.

A total line is generated at the end of the display containing the total number of free tracks or blocks.

The following fields are displayed in the VOLID list:

VOLID	Volume ID.
CUU	Device address.
DEVICE TYPE	Hardware device type (mmmm-tts) (S) if shared DASD FBA/CKD/ECKD indication.
TRKS-BLKS	Total device capacity in FBA blocks or CKD tracks.
CYLS	Total number of cylinders.
TRK/CYL	Number of tracks per cylinder.
TRK SIZE	Maximum track size (in bytes for CKD, blocks for FBA).
MAX REC	Maximum CKD record size or FBA block size.
SYSNNN	SYSnnn assigned in BIM-EDIT partition. Will be blank if not yet assigned.

When Valid

Your system must be on VSE/ESA 2.1 or above.

Use in a Procedure

Return Codes:

OK	Successful
NF	Requested volume not found.
SV	Error returned from VTOC interface.

Examples

Create a display session for the VTOC for FBA volume DSKC02.

```
=> vtoc dskc02
```

```
=>
DISP -> vtoc dskc02
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----|-----8-----|-----9-----|-----10-----|-----11-----|-----12-----
-- TOP OF DISPLAY --
VTOC DISPLAY FOR VOLUME: DSKC02          DATE: 11/30/2000  TIME: 14:50:17
                                TYPE: 9336-10  FBA          XT NBR. OF  FILE -- STARTING --- --- ENDING ---  CREATE  EXPIRE
                                SQ  BLOCKS  TYPE          BLOCK          BLOCK          DATE    DATE
-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
*** VOL1 AND IPL ***
VSE.ESA.241.SYSRES                2  SYST                0                1
                                49998  UNDF                2                49999  2000/245  2099/365
*** FREE SPACE ***
                                164000
VSE.ESA.241.SYSGEN1              32000  UNDF              214000  245999  2000/246  2099/365
GSS.SYSSVIO.PDS                   40000  SAM                246000  285999  1998/069  2099/365
GSS.SYSSMON.PDS                   24000  SAM                286000  309999  1998/069  2099/365
GSS.SYSSARC.PDS                    4000  SAM                310000  313999  1998/069  2099/365
GSS.SYSSCPR.PDS                    9000  SAM                314000  322999  1998/069  2099/365
GSS.SYSSVET.PDS                   15000  SAM                323000  337999  2000/059  2099/365
GSS.SYSSLOG.PDS                    5000  SAM                338000  342999  1998/069  2099/365
*** FREE SPACE ***
                                260000
ODIS.SYSPCH.WORK                   8000  *SAM                603000  610999  1997/136  1997/136
*** FREE SPACE ***
                                31720
DOS.PAGING.FILE.100F07569221      0  267456  UNDF              642720  910175  2000/326  2099/366
*** FREE SPACE ***
                                24
VSE.ESA.241.PRD2                  170000  UNDF              910176  910199
DOS.PAGING.FILE.100F07569221      1  183104  UNDF              1080200  1263303  2000/326  2099/366
*** FREE SPACE ***
                                49416
VSE.ESA.241.PRD1                  360145  SAM              1312720  1672864  2000/175  2020/365
*** VTOC EXTENT ***
                                16  SYST              1672865  1672880

TOTAL AMOUNT OF FREE-SPACE ON THIS VOLUME: 505160 BLOCKS
-- END OF DISPLAY --
```

Create a display session for the VTOC for ECKD volume DSK50C.

```
=> vtoc dsk50c
```

```
=>
DISP -> VTOC      DSK50C
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----|-----8-----|-----9-----|-----10-----|-----11-----|-----12-----
-- TOP OF DISPLAY --
VTOC DISPLAY FOR VOLUME: DSK50C          DATE: 11/30/2000  TIME: 14:55:58
                                TYPE: 9345-04  ECKD          XT NBR. OF  FILE -- STARTING --- --- ENDING ---  CREATE  EXPIRE
                                SQ  TRACKS  TYPE          CCCC.HH  TRACK  CCCC.HH  TRACK  DATE    DATE
-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
*** VOL1 AND IPL ***
                                1  SYST                0.00      0                0.00      0
*** FREE SPACE ***
                                1364
Z99999994.VSAMDSPC.TAEA98B0.T9718A0E  30  VSAM                91.00    1365  92.14    1394  1997/135  2099/366
Z99999992.VSAMDSPC.TAEA98B0.TDC305AE  9000  VSAM                93.00    1395  692.14  10394  1997/135  2099/366
BIMEDITT.CKDTEST.BIFLIB.53A+        0  600  DAM                693.00  10395  732.14  10994  2000/102  2099/365
EPIC.VSE.CATALOG                     30  UNDF                733.00  10995  734.14  11024  1998/247  2099/366
EPIC.VSE.RECORDER                     15  UNDF                735.00  11025  735.14  11039  1998/247  2099/366
*** FREE SPACE ***
                                105
BIMEDITT.CKDTEST.BIFLIB.53A+        1  300  DAM                743.00  11145  762.14  11444  2000/102  2099/365
BIMEDITG.54A.99SESS.LIBRARY          0  300  DAM                763.00  11445  782.14  11744  2000/319  2099/365
DRJ.PDS.CENTAL.INS.TEST              2000  SAM                783.00  11745  916.04  13744  1998/191  2099/365
BIMEDITT.CKDTEST.BIFLIB.53A+        2  2000  DAM                916.05  13745  1049.09  15744  2000/102  2099/365
*** FREE SPACE ***
                                3
BIMEDITT.CKDTEST.BIFLIB.53A+        3  400  DAM                1049.10  15745  1049.12  15747
BIMEDITG.54A.99SESS.LIBRARY          1  100  DAM                1076.08  16148  1083.02  16247  2000/319  2099/365
*** FREE SPACE ***
                                27
DQ.TEST.INPUT.FILE                   15  *SAM                1085.00  16275  1085.14  16289  2000/056  2000/056
DQ.TEST.OUTPUT.FILE                  15  *SAM                1086.00  16290  1086.14  16304  2000/056  2000/056
BIMEDITT.CKDTEST.BIFLIB.53A+        4  4080  DAM                1087.00  16305  1358.14  20384  2000/102  2099/365
Z99999992.VSAMDSPC.TAF13111.T2E7173E  9000  VSAM                1359.00  20385  1958.14  29384  1997/219  2099/366
*** FREE SPACE ***
                                2940
*** VTOC EXTENT ***
                                15  SYST                2155.00  32325  2155.14  32339

TOTAL AMOUNT OF FREE-SPACE ON THIS VOLUME: 4439 TRACKS
```


Create a display session for all VOLIDs defined to VSE.

```
=> vtoc
```

```
=>
DISP  -> vtoc                                SESS=A 1( 1) LINE=    0(   16)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----
*====* -- TOP OF DISPLAY --
*====* VOLID CUU DEVICE TYPE TRKS-BLKS CYLS TRK/CYL TRK SIZE MAX REC SYSNNN
*====* -----
*====* BIMVIO 266 3370-00 FBA 49104 66 12 62 512 SYS098
*====* BIMVI2 267 3370-00 FBA 20088 27 12 62 512 SYS096
*====* VSESHR 311 3390-0AS ECKD 50085 3339 15 58786 56664 SYS098
*====* DSK508 508 9345-04 ECKD 32340 2156 15 48280 46456 SYS095
*====* DSK509 509 9345-04 ECKD 32340 2156 15 48280 46456 SYS097
*====* DSK50A 50A 9345-04 ECKD 32340 2156 15 48280 46456 SYS094
*====* DSK50B 50B 9345-04 ECKD 32340 2156 15 48280 46456 SYS093
*====* DSK50C 50C 9345-04 ECKD 32340 2156 15 48280 46456 SYS061
*====* DSK50D 50D 9345-04 ECKD 32340 2156 15 48280 46456 SYS062
*====* DSK50E 50E 9345-04 ECKD 32340 2156 15 48280 46456 SYS064
*====* DSK50F 50F 9345-04 ECKD 32340 2156 15 48280 46456 SYS063
*====* DSKC00 C00 9336-10 FBA 1672881 2153 7 111 512 SYS041
*====* DSKC01 C01 9336-10 FBA 1672881 2153 7 111 512 SYS070
*====* DSKC02 C02 9336-10 FBA 1672881 2153 7 111 512 SYS018
*====* DSKC03 C03 9336-10 FBA 1672881 2153 7 111 512 SYS099
*====* -- END OF DISPLAY --
```

&(Ampersand)

Use the ampersand (&) before a command to cause the ampersand and all commands following it to redisplay on the command line after the transaction. The command(s) may then be executed again simply by pressing the ENTER key. The following conventions apply:

- More than one command may follow the ampersand (separated by semicolons).
- All will be executed during the transaction.
- The ampersand and command string following the ampersand will redisplay on the command line.

Example

The LOCATE command positions the current line at the first line having an occurrence of the user-supplied character string. Placing the repeat symbol (&) before the command will allow searching through the session text, stopping at each line where the character string is found, each time the ENTER key is pressed:

```
=> &l iolength
```

=(Equal Sign)

Use the equal sign to cause the last command entered to be processed again.

Equal sign is useful with formatting commands such as FORMAT, SEPARATE, or SHIFT which need to be executed in several places on a screen. You can use the LCA / to reposition in the session without changing the most recent command.

Example**EDIT session before the SHIFT command**

```
=> shift +5 zone=15-*
EDIT 1283.CPACT                      SESS=A 1( 1) LINE= 17( 19)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*====* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*====* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
-- END OF MEMBER --
```

**EDIT session after the SHIFT command, before the =
(Note the / used to reposition)**

```
=> =
EDIT 1283.CPACT                      SESS=A 1( 1) LINE= 17( 19)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* SEE THE QUICK      BROWN FOX JUMP OVER THE LAZY BROWN DOG
*====* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
/====* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
-- END OF MEMBER --
```

EDIT session after the =

```
=>
EDIT 1283.CPACT                      SESS=A 1( 1) LINE= 19( 19)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* SEE THE QUICK      BROWN FOX JUMP OVER THE LAZY BROWN DOG
-- END OF MEMBER --
```

?(Question Mark)

Use the question mark (?) to recall the previous command entered to be displayed on the command line. Repeating the enter key, with no intervening activity, will recall the next previous, etc.

The selected command can then be shifted and modified (if desired) and processed by depressing the ENTER key.

Use the double question mark (??) to create a display session of your internal command stack.

Use the question mark (?), on the LCA of the command stack display session, to select a command for execution. The line can be modified if the command stack display session is updateable (MMPDSCCTL set to 1 or 2).

Command stacking considers commands only from an 'Enter Key' and then only if they are greater than two characters in length. If you use the command stack as a procedure developing aid, don't abbreviate the commands being entered.

Example procedure, to reverse the order of a command stack display selection, useful for procedure development.

```
SET PPDCOND, 2
ST ,CL
B
FOR
BLOCK
    ST 1,AP
    U
    IF SIBRETC D EQ OK
        LOOP
    Q
ENDBLOCK
LI $STACK
EXIT
```

Examples

Display the last command:

```
=> ?
```

Create a Display session of the command stack.

```
=> ??
```

<(Less Than)

Use the less than character to switch the "primary screen" (the one to which the command line applies) between the two logical screens shown in split screen mode. Its use allows commands to be processed against both logical screens with one ENTER key.

Examples

Change to the other logical screen:

```
=> <
```

Find the string "lost" in the other logical screen:

```
=> <;find lost
```

Position both logical screens down 9 lines each time the ENTER is pressed:

```
=> &down 9;<;down 9
```

Chapter 5. LCA Commands

The line control area (LCA) is used for commands associated with the text lines displayed on the screen. You may use the LCA mode of the SCREEN command to control whether it is on the left side or right side of the text display area and whether it displays at all. The LCA is prefilled with the five characters (*==*) unless you set STAMP mode ON with the SCREEN command.

You can enter LCA commands anywhere in the LCA area. An LCA command applies to the text line next to it and possibly to lines following. Some LCA commands act against the text line itself (example: the LCA D command deletes the line next to it). Others act against an object described by the text line (example: the LCA P command purges the member described by the text line next to it on a LIBRARY display).

The following LCA commands are provided. The equivalent command line entry is shown in the right hand column.

LCA Commands		Equiv. Cmd
ALTERP	Alter POWER job entry attributes.	
A	Insert blank text lines after this text line.	LADD
B	Insert text lines from \$STACK before this text line	GET
C, CC	Copy these text lines to \$STACK.	STACK
D, DD	Delete these text lines.	DELETE
E, EE	Edit this member.	EDIT
G	Merge text lines from \$STACK with these text lines.	MERGE
H, HH	Put these POWER job entries or JES data sets on hold	HOLD
I	Insert text lines from \$STACK after this text line.	GET
J, JJ	Merge text lines from \$STACK with these text lines.	MERGE
K, KK	Append these text lines to \$STACK.	STACK ,AP
L, LL	List text of this member, message, etc.	LIST, etc.
M, MM	Move - copy these text lines to \$STACK and delete.	STACK;DEL
N, NN	Append these text lines to \$STACK and delete.	STACK;DEL
P, PP	Delete these members, messages, etc.	PURGE, etc.
Q, QQ	Display/alter these members, libraries.	FALTER, etc
R, RR	Take these POWER job entry / JES data sets off hold	RELEASE
S, SS	Process this member - submit, compile or execute.	PROCESS
T, TT	Display text of this member, etc.	DISPLAY
U, UU	Translate these text lines to upper case.	UPPERCAS
W, WW	Translate these text lines to lower case.	LOWERCAS
X, XX	Display list of JES or POWER datasets within a job.	INQUIREP
Y	Attach to this library.	ATTACH
", ""	Duplicate text lines.	DUP
\	Position this text line to the bottom of the screen.	
/	Position session to this text line.	POSITION
<, <<	Shift these text lines left.	SHIFT
>, >>	Shift these text lines right.	SHIFT
+, ++	Center these text lines.	CENTER
(, ((Left justify these text lines.	JUSTIFYL

) ,))	Right justify these text lines.	JUSTIFYR
---------	---------------------------------	----------

BIM-EDIT supports Site Defined LCA Commands. These LCA commands are two-characters in length, and begin with the letter 'Z'. Contact the person at your site responsible for BIM-EDIT to determine if any Site LCA Commands are available for your use.

This chapter provides a comprehensive description of each LCA command. The commands are presented in the above order.

The remainder of this introduction covers some common LCA command format issues.

Repeat Count LCA Commands

You can follow most LCA commands (including Site Defined LCA Commands) by digits to indicate how many times the command is to be repeated. The repeat count may be 1 to 9999. If the repeat count is not entered, a value of 1 is assumed. Examples of this usage would be A5 to insert 5 blank lines or D99 to delete this line and 98 lines following.

Bracket LCA Commands

You can enter most LCA commands as a pair of "brackets" by doubling the command letter (for example, CC, MM, KK, LL). One bracket is entered on the first line to which the command is to apply and the other is entered on the last line. The command affects all the lines between and including the brackets. In practice, you will enter bracket commands in three different ways:

- Enter both the opening and closing brackets on the same screen. This method can only be used when the text can be contained on one screen, limiting the range to 21 to 40 lines, depending upon your terminal model.
- Enter the opening bracket, use positioning commands (FORWARD, LOCATE, UP, etc.) to locate the closing position, and then enter the closing bracket. Any number of lines can be affected. While there is one bracket entered, it is said to be "pending". The pending bracket will be indicated at the top of the LCA and special rules apply to the entry of other LCA commands.
- In split-screen mode with both logical screens viewing the same session, enter the opening bracket on one logical screen and the closing bracket command on the other. Any number of lines can be affected.

Site Defined LCA Commands can also be entered as a pair of "brackets" by repeated the second character of the LCA command. For example, if your site has defined "ZA" as an LCA command, you would specify "ZAA" as the "bracket" format.

Transferring Lines

\$STACK is used to temporarily hold text lines when they are transferred using LCA commands. Each user has a single \$STACK; text lines copied into \$STACK remain there until they are overwritten by a subsequent command. Thus, lines in \$STACK may be inserted several times into several different screens or sessions, if desired.

The C, CC, M, and MM commands initialize \$STACK by clearing any lines which were there from a previous command, and then move copies of the indicated text lines to it. The K, KK, N, and NN commands place copies of the indicated text line(s) after those already in \$STACK. The I command inserts lines from \$STACK into the session after the line indicated. The B command inserts lines from \$STACK into the session before the line indicated.

To move or copy text from one place to another:

1. If necessary, open and position a session to the lines to be transferred.
2. Enter the C, CC, M, or MM LCA commands to indicate the lines to be transferred. (If you are collecting lines from multiple sources, enter the K, KK, N, or NN commands.)
3. If necessary, open and position an EDIT session to the destination.
4. Enter the LCA I or B command to indicate where to insert the transferred lines.
5. If the same lines are to be multiply inserted, repeat steps 3 and 4.

If the lines to be transferred and the destination are on a single screen (which is common), you may indicate both the source lines and destination in one transmission.

Multiple LCA Commands on a Screen

LCA commands may be entered on several lines prior to transmission.

All C, CC, M, MM, K, KK, N or NN commands are processed before any other commands (most notably I, B, J or JJ).

If there are multiple C, CC, M, MM, K, KK, N, or NN commands on the screen, the commands are processed beginning at the command closest to the top of the screen, then the next command down and so forth. If the lines are later transferred from \$STACK into a session, they will appear in this top down order.

Multiple LCA commands can be entered on one line. For example, entering "c/==*" causes the line to be copied to \$STACK and causes the line to become the current line.

If any LCA command is in error, none of the LCA commands will be processed and an error message will display on the information line. In this case, all LCA commands will be redisplayed with highlighting and the cursor will be positioned at the command where BIM-EDIT detected the error. You can correct the LCA command error and retransmit the screen.

A(n)

Add blank line(s).

Immediately following the line with the A command, n blank lines will be added. The default value for n is 1. Upon reply, the cursor will be positioned at the first new blank line in the edit area.

Example

EDIT session before A commands

```
=>
EDIT  NA32.SYCREN                      SESS=A 1( 1) LINE=      1(      5)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*      DCL
a====*      CF ALLOC BAL                  PIC 'ZZZZZZ9V.99',
*====*      CF UNALLOC BAL                PIC 'ZZZZZZ9V.99',
a2====*      CF ORD BAL                   PIC 'ZZZZZZ9V.99',
*====*      CF CR BAL                     PIC 'ZZZZZZ9V.99',
-- END OF MEMBER --
```

EDIT session after A commands

```
=>
EDIT  NA32.SYCREN                      SESS=A 1( 1) LINE=      1(      8)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*      DCL
*====*      CF ALLOC BAL                  PIC 'ZZZZZZ9V.99',
*====*
*====*      CF UNALLOC BAL                PIC 'ZZZZZZ9V.99',
*====*      CF ORD BAL                   PIC 'ZZZZZZ9V.99',
*====*
*====*      CF CR BAL                     PIC 'ZZZZZZ9V.99',
-- END OF MEMBER --
```

B(n)

Insert the contents of \$STACK n times immediately before this line. If n is not specified, the \$STACK contents will be inserted once.

This command is used after lines have been loaded into \$STACK with the C, CC, K, KK, M, MM, N and NN commands.

Since only copies are placed into the EDIT session, the lines are available for insertion elsewhere, until \$STACK is erased with another C, CC, M, or MM command. Upon reply, the cursor will be positioned on the first line inserted.

Example

The line with the C command is copied into \$STACK, and that line is copied into the member before each line with the B command.

EDIT session before the C and B commands

```
=>
EDIT 4231.XM25                                SESS=A 1( 1) LINE= 21( 234)
-----1-----2-----3-----4-----5-----6-----7--
c===*      CALL CHECK;                        /* CHECK DATA FOR ERRORS */
*===*      TXC COMPANY = C COMPANY;
b===*      READ FILE (TXMFL) INTO (TXC_XRCD) KEY (TXC_KEY);
*===*      IF TXM EOF THEN
*===*          CALL ERROR;
b===*      TXM MBR = 'EXTA';
*===*      READ FILE (TXMFL) INTO (TXM_RCD) KEY (TXM_KEY);
*===*      S20 TYPE = TXM TYPE;
*===*      S20 NAME = TXM NAME;
```

EDIT session after the C and B commands

```
=>
EDIT 4231.XM25                                SESS=A 1( 1) LINE= 21( 236)
-----1-----2-----3-----4-----5-----6-----7--
*===*      CALL CHECK;                        /* CHECK DATA FOR ERRORS */
*===*      TXC COMPANY = C COMPANY;
*===*      CALL CHECK;                        /* CHECK DATA FOR ERRORS */
*===*      READ FILE (TXMFL) INTO (TXC_XRCD) KEY (TXC_KEY);
*===*      IF TXM EOF THEN
*===*          CALL ERROR;
*===*      CALL CHECK;                        /* CHECK DATA FOR ERRORS */
*===*      TXM MBR = 'EXTA';
*===*      READ FILE (TXMFL) INTO (TXM_RCD) KEY (TXM_KEY);
```

C(n),CC

Copy specified number of lines from the session into \$STACK.

If the C form is used, n specifies the number of lines to copy. If n is not specified, one line will be copied. If the CC form is used, the CC command is entered on both the starting and ending lines of the group of lines to copy.

The top-most C, CC, M, or MM command on the screen will clear \$STACK.

If lines in addition to those already in \$STACK are to be added, use the K or KK commands.

Example

EDIT session before C,CC commands

```
=>
EDIT 2458.MSRM250                      SESS=A 1( 1) LINE= 21( 234)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
c2==*   GET EDIT (CARD) (A(80));
*==*   TXC COMPANY = C COMPANY;
*==*   READ FILE (TXMFL) INTO (TXC XRCD) KEY (TXC KEY);
*==*   IF TXM_EOF THEN
*==*       DO;
cc==*       PUT SKIP EDIT
*==*           (CARD) (A);
cc==*       PUT SKIP;
*==*   END;
```

\$STACK after C,CC commands

```
=>
LIST $STACK                      SESS=A 2( 2) LINE= 0( 5)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==* -- TOP OF $STACK --
*==*   GET EDIT (CARD) (A(80));
*==*   TXC COMPANY = C COMPANY;
*==*       PUT SKIP EDIT
*==*           (CARD) (A);
*==*       PUT SKIP;
*==* -- END OF $STACK --
```

D(n),DD

Delete specified number of lines.

If the D form is used, n specifies the number of lines to delete. If n is not specified, one line will be deleted. If the DD form is used, DD must be entered on both the starting and ending lines of the group of lines to be deleted.

Example**EDIT session before D,DD commands**

```
=>
EDIT HGX3.HGRM200                                SESS=A 1( 1) LINE= 21( 234)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
d2==*      IF SYSIN EOF THEN
*====*      LEAVE;
*====*      READ FILE (TXMFL) INTO (TXC XRCD) KEY (TXC KEY);
*====*      IF TXM EOF THEN
*====*      DO;
*====*          PUT SKIP EDIT (CARD) (A);
dd==*          PUT SKIP EDIT
*====*              ('* COMPANY ') (A)
*====*              (C COMPANY) (A)
dd==*              (' DOES NOT EXIST **') (A);
*====*      END;
```

EDIT session after D,DD commands

```
=>
EDIT HGX3.HGRM200                                SESS=A 1( 1) LINE= 21( 228)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*      READ FILE (TXMFL) INTO (TXC XRCD) KEY (TXC KEY);
*====*      IF TXM EOF THEN
*====*      DO;
*====*          PUT SKIP EDIT (CARD) (A);
*====*      END;
*====*      TXM RC = 'M';
*====*      TXM LIB = TXC LIB;
*====*      TXM MBR = 'EXTA';
*====*      READ FILE (TXMFL) INTO (TXM RCD) KEY (TXM KEY);
*====*      S20 TYPE = TXM TYPE;
*====*      S20 ATTR = TXM ATTR;
```

E(n),EE

Create an EDIT session of a member.

EDIT sessions are created by entering the E command on a LIBRARY display.

Example

Edit member BICALTL:

```
=>
DISP -> library bic,det                      SESS=A 1( 1) LINE=      0( 174)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF DISPLAY --
*====* LIBRARY=ED51A
*====* -----
```

MEMBER	TYPE	ATTR	--- LINES ---		DATE	
			TEXT	AUDIT	CREATE	UPDATE
==== BICADDX	ASM	NL	49	0	03/05/1995	
==== BICALTL	ASM	NL	71	0	03/05/1995	
==== BICALTM	ASM	NL	112	114	03/05/1995	06/05/1995
==== BICALTP	ASM	NL	162	518	03/05/1995	06/23/1995
==== BICALTS	ASM	NL	70	8	03/05/1995	03/28/1995
==== BICALTU	ASM	NL	117	215	03/05/1995	05/25/1995
==== BICATTA	ASM	NL	66	51	03/05/1995	06/12/1995
==== BICATTX	ASM	NL	68	0	03/05/1995	
==== BICAUCL	ASM	NL	99	0	03/05/1995	
==== BICAUDF	ASM	NL	39	0	03/05/1995	
==== BICAUDI	ASM	NL	42	0	03/05/1995	
==== BICAURL	ASM	NL	242	191	03/05/1995	05/29/1995
==== BICAURP	ASM	NL	112	64	03/05/1995	05/18/1995
==== BICAUSM	ASM	NL	222	17	03/05/1995	05/18/1995
==== BICAUTH	ASM	NL	51	0	06/16/1995	06/16/1995

Merge \$STACK text lines with lines from the current EDIT session.

The J,JJ commands differ from the G command in that the J,JJ commands specify how many lines to update whereas the G command always updates the exact number of lines that are held in \$STACK.

Session lines are updated by overlaying non-blank text from \$STACK lines onto the corresponding session lines. The overlay occurs for a given line position only if the position is blank in the session line. Session lines are updated until \$STACK is exhausted. For example, if \$STACK contains 12 lines, 12 session lines will be updated.

Example

\$STACK before and after G command

```
=>
LIST $STACK                                SESS=A 2( 2) LINE=    0(    3)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*  -- TOP OF $STACK --
*====*          Cotter pin. Prevents the wheel from falling
*====*          off the motor shaft. Approximately three
*====*          inches in length. Made of 14 ga. copper wire.
*====*  -- END OF $STACK --
```

EDIT session before G command

```
=>
EDIT 5624.MATERIAL                          SESS=A 1( 2) LINE=    8(   42)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* |
g====* | X387 |
*====* | |
*====* | |
*====* | |
*====* | X388 |
```

EDIT session after G command

```
=>
EDIT 5624.MATERIAL                          SESS=A 1( 2) LINE=    8(   42)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* |
*====* | X387 | Cotter pin. Prevents the wheel from falling |
*====* | | off the motor shaft. Approximately three |
*====* | | inches in length. Made of 14 ga. copper wire. |
*====* | |
*====* | X388 |
```

H(n),HH

Hold a POWER job entry or a JES job or data set.

A POWER job entry or a JES job or data set are held by entering the H command on a LIBRARYP display.

VSE Example

On VSE, hold a group of POWER LST jobs:

```
=>
DISP -> libraryp lst                      SESS=A 1( 1) LINE=      0(   18)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----
*====* -- TOP OF DISPLAY --
*====* ----- POWER JOB ----- P D C S   CARDS/ CPY FORM   ORIGIN/   USER INFO
*====* QUE      NAME    NUMBER SG          PAGES          DEST
*====* -----
*====* LST BIMWNC22 11040    3 * A         132    1          BDB
hh==* LST BIMWNC  11195    3 D A         150    1          BDB
*====* LST BIMWNX   11197    3 D A          22    1          BDB
*====* LST CICDEF   11890    3 D A          15    1          CIC2
*====* LST AMA3100C 11993    3 D A          61    1          CIC2
hh==* LST VSDFSND  12144    3 D A           9    1          JIMF
*====* -- END OF DISPLAY --
```

MVS Example

On MVS, hold a group of JES jobs and data sets:

```
=>
DISP -> libraryp                      SESS=A 1( 1) LINE=      0(   17)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----|-----
*====* -- TOP OF DISPLAY --
*====* ----- INPUT QUEUE -----
*====* JOB ---- T C RMT PR POS EXEC STAT
*====* NAME  NUMBER Y L NODE TY      SYS  -US
*====* -----
*====* ----- HELD QUEUE -----
*====* JOB ---- T C RMT FORM FCB   CREATE   LINES  ROOM   PROGRAMMER
*====* NAME  NUMBER Y L NODE ID      DATE      NMBR
*====* -----
*====* ----- OUTPUT QUEUE -----
*====* JOB ---- T C RMT FORM FCB   CREATE   LINES  PROC-  PRI  STAT
*====* NAME  NUMBER Y L NODE ID      DATE      ESSED  -US
*====* -----
hh==* BIM001B   8075 J X    1 STD   ****  08/01/1996    90    0    9
hh==* BICEDTD   8083 J X    1 STD   ****  08/01/1996   347    0    9
*====* -- END OF DISPLAY --
```

I(n)

Insert the contents of \$STACK n times immediately after this line. If n is not specified, the \$STACK contents will be inserted once.

This command is used after lines have been loaded into \$STACK with the C, CC, K, KK, M, MM, N, and NN commands.

Since only copies are placed into the EDIT session, the lines are available for insertion elsewhere, until \$STACK is erased with another C, CC, M, or MM command. Upon reply, the cursor will be positioned on the first line inserted.

Example

The line with the C command is copied into \$STACK, and that line is copied into the member after each line with the I command.

EDIT session before the C and I commands

```
=>
EDIT 4231.XM25                                SESS=A 1( 1) LINE= 21( 234)
-----1-----2-----3-----4-----5-----6-----7--
c==== CALL CHECK;                               /* CHECK DATA FOR ERRORS */
*==== TXC COMPANY = C COMPANY;
i==== READ FILE (TXMFL) INTO (TXC_XRCD) KEY (TXC_KEY);
*==== IF TXM EOF THEN
*==== CALL ERROR;
i==== TXM MBR = 'EXTA';
*==== READ FILE (TXMFL) INTO (TXM_RCD) KEY (TXM_KEY);
*==== S20 TYPE = TXM TYPE;
*==== S20 NAME = TXM NAME;
```

EDIT session after the C and I commands

```
=>
EDIT 4231.XM25                                SESS=A 1( 1) LINE= 21( 236)
-----1-----2-----3-----4-----5-----6-----7--
*==== CALL CHECK;                               /* CHECK DATA FOR ERRORS */
*==== TXC COMPANY = C COMPANY;
*==== READ FILE (TXMFL) INTO (TXC_XRCD) KEY (TXC_KEY);
*==== CALL CHECK;                               /* CHECK DATA FOR ERRORS */
*==== IF TXM EOF THEN
*==== CALL ERROR;
*==== TXM MBR = 'EXTA';
*==== CALL CHECK;                               /* CHECK DATA FOR ERRORS */
*==== READ FILE (TXMFL) INTO (TXM_RCD) KEY (TXM_KEY);
```


J(n),JJ

Merge \$STACK text lines with lines from the current EDIT session.

The J,JJ commands differ from the G command in that the J,JJ commands specify how many lines to update whereas the G command always updates the exact number of lines that are held in \$STACK.

Session lines are updated by overlaying non-blank text from \$STACK lines onto the corresponding session lines. The overlay occurs for a given line position only if the position is blank in the session line.

If the J form is used, n specifies the number of lines to update. If n is not specified, one line is updated. If the JJ form is used, JJ must be entered on both the starting and ending line of the group of lines to be updated. If more lines are specified than are held in \$STACK, \$STACK will be re-used as needed. Upon reply, the cursor will be positioned on the top-most line updated.

Example

\$STACK before and after JJ command

```
=>
LIST $STACK                                SESS=A 2( 2) LINE=    0(    1)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==*  -- TOP OF $STACK --
*==*  |                     |
*==*  -- END OF $STACK
```

EDIT session before JJ command

```
=>
EDIT 5624.MATERIAL                          SESS=A 1( 2) LINE=    8(   42)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==*  |-----|
*==*  | X387      Cotter pin. Prevents the wheel from falling |
*==*  |           off the motor shaft. Approximately three   |
*==*  |           inches in length. Made of 14 ga. copper wire. |
*==*  |-----|
```

EDIT session after JJ command

```
=>
EDIT 5624.MATERIAL                          SESS=A 1( 2) LINE=    8(   42)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==*  |-----|
*==*  | X387      | Cotter pin. Prevents the wheel from falling |
*==*  |           | off the motor shaft. Approximately three   |
*==*  |           | inches in length. Made of 14 ga. copper wire. |
*==*  |-----|
```

K(n),KK

Copy lines into \$STACK without first deleting the contents of \$STACK.

If the K form is used, n specifies the number of lines to copy. If n is not specified, one line is copied. If the KK form is used, KK must be entered on both the starting and ending line of group of lines to be copied.

After copying the first lines into \$STACK with the C, CC, M, or MM commands, you can add other lines using the K or KK commands. These lines may be from the same session or from any number of other sessions. The I or B command will copy all the lines in \$STACK into an EDIT session.

Example**\$STACK before K command**

```
=>
LIST  $STACK                                SESS=A 3( 3) LINE=    0(    2)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF $STACK --
*====* Logging On and Logging Off
*====* Screen Layout
*====* -- END OF $STACK --
```

EDIT session before and after K command

```
=>
EDIT  ED51A.BIZUSRF                        SESS=A 2( 2) LINE= 1044( 13397)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* .* -hlp-
*====* .*****
k====* Online Help
*====*
*====* The BIM-EDIT online help facility is accessed using the HELP command.
*====*
*====* At all times, the HELP command is available for immediate access
*====* reference manual descriptions of the commands, the LCA commands,
```

\$STACK after K command

```
=>
LIST  $STACK                                SESS=A 3( 3) LINE=    0(    3)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF $STACK --
*====* Logging On and Logging Off
*====* Screen Layout
*====* Online Help
*====* -- END OF $STACK --
```

L(n),LL

Create a LIST session of a member, a LISTI session of an ICCF member, a LISTP session of a POWER job entry or JES data sets, a LISTD session of a VSE sublibrary member or MVS PDS member, or OPEN a message.

LIST sessions are created by entering the L command on a LIBRARY display. LISTI sessions are created by entering the L command on a LIBRARYI display. LISTP sessions are created by entering the L command on a LIBRARYP display. LISTD sessions are created by entering the L command on a LIBRARYD display. A message is OPENed by entering the L command on a LIBRARYQ display.

Example

List member \$SYS.UDP.BIMISCAN

```
=>
DISP -> library $sys.udp.,det                      SESS=A 1( 1) LINE=      0(   52)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==== -- TOP OF DISPLAY --
*==== LIBRARY=$SYS.UDP
*====
*==== MEMBER          TYPE      ATTR      --- LINES ---      DATE -----
*====              TEXT  AUDIT      CREATE      UPDATE
*====-----
*==== ALLMEMS          PROC              63          0 06/09/1991 06/09/1991
*==== BIMCICS          COBOL      CL       31          0 04/12/1990 04/12/1990
1==== BIMISCAN          JCL              22          0 12/05/1991 12/06/1991
*==== BIMPRNT          DATA      504          0 08/09/1989 05/23/1990
*==== BULKHELP          TEXT       16          0 11/18/1991 11/18/1991
```

Resulting session

```
=>
LIST $SYS.UDP.BIMISCAN                      SESS=A 2( 2) LINE=      0(   22)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==== -- TOP OF MEMBER --
*==== * $$ JOB JNM=ICCFSCAN,CLASS=0,DISP=D
*==== * $$ LST DISP=H,CLASS=Q
*==== // JOB ICCFSCAN
*==== // TLBL MTOUT,'ICCF LIB BACKUP'
*==== // ASSGN SYS010,182          ICCF LIBRARY BACKUP TAPE
*==== * FORMAT: PUND=PROG1234          PUNCH PROGRAM
*==== * FORMAT: LIB=NNN          LIBRARY TO BE SEARCHED OR ALL
*==== * FORMAT: SCAN=NN,'ARG'          SCAN FORMAT
```

M(n),MM

Functions as if the C (COPY) command had been entered for the line(s), and then a D (DELETE) command had been entered for the same lines. Lines are copied to \$STACK, and then deleted from the session.

If the M form is used, n specifies the number of lines to copy and delete. If n is not specified, one line is copied and deleted. If the MM form is used, MM must be entered on both the starting and ending line of the group of lines to be copied and deleted.

Example**EDIT session before M,MM commands**

```
=>
EDIT 5624.TMXA356                      SESS=A 1( 1) LINE= 224( 231)
-----1-----2-----3-----4-----5-----6-----7--
m==== GET EDIT (CARD) (A(80));
*==== TXC COMPANY = C COMPANY;
mm==== READ FILE (TXMFL) INTO (TXC_XRCD) KEY (TXC_KEY);
*==== IF TXM EOF THEN
*==== DO;
mm==== TXM EOF = '0'B;
*==== PUT SKIP EDIT (CARD) (A);
*==== END;
```

EDIT session after M,MM commands

```
=>
EDIT 5624.TMXA356                      SESS=A 1( 1) LINE= 224( 226)
-----1-----2-----3-----4-----5-----6-----7--
*==== TXC COMPANY = C COMPANY;
*==== PUT SKIP EDIT (CARD) (A);
*==== END;
-- END OF MEMBER --
```

\$STACK after M commands

```
=>
LIST $STACK                          SESS=A 2( 2) LINE= 0( 5)
-----1-----2-----3-----4-----5-----6-----7--
*==== -- TOP OF $STACK --
*==== GET EDIT (CARD) (A(80));
*==== READ FILE (TXMFL) INTO (TXC_XRCD) KEY (TXC_KEY);
*==== IF TXM EOF THEN
*==== DO;
*==== TXM EOF = '0'B;
*==== -- END OF $STACK --
```

N(n),NN

Copy lines to \$STACK without first deleting its contents. After copying to \$STACK, the lines are deleted from the session. If the N form is used, n specifies the number of lines to copy and delete. If n is not specified, one line is processed. If the NN form is used, NN must be entered on both the starting and ending line of group of lines to be processed.

After copying lines into \$STACK with the C, CC, M, or MM commands, you can use the N and NN commands to add other lines from the same or other sessions. The I or B command will copy all the lines in \$STACK into an EDIT session.

Example

\$STACK before N command

```
=>
LIST $STACK                                SESS=A 3 ( 3) LINE=    0 (    1)
----|----1----|----2----|----3----|----4----|----5----|----6----|----7--
*====* -- TOP OF $STACK --
*====* 100 FORMAT(4A4)
*====* -- END OF $STACK --
```

EDIT session before N command

```
=>
EDIT MECH.HCCALC                          SESS=A 2 ( 2) LINE=   43 (   205)
----|----1----|----2----|----3----|----4----|----5----|----6----|----7--
*====*      WRITE(IOUT,101),ROWNO(I), (TRNSFR(I,J),J=1,7)
n====* 101 FORMAT(I8,X1,7F12.3)
*====* 900 CONTINUE
```

EDIT session after N command

```
=>
EDIT MECH.HCCALC                          SESS=A 2 ( 2) LINE=   43 (   204)
----|----1----|----2----|----3----|----4----|----5----|----6----|----7--
*====*      WRITE(IOUT,101),ROWNO(I), (TRNSFR(I,J),J=1,7)
*====* 900 CONTINUE
```

\$STACK after N command

```
=>
LIST $STACK                                SESS=A 3 ( 3) LINE=    0 (    2)
----|----1----|----2----|----3----|----4----|----5----|----6----|----7--
*====* -- TOP OF $STACK --
*====* 100 FORMAT(4A4)
*====* 101 FORMAT(I8,X1,7F12.3)
*====* -- END OF $STACK --
```

P(n),PP

Purge a library, member, POWER job entry or JES job or data sets, VSE sublibrary member or MVS PDS member, or message.

Libraries are purged by entering the P command on a LIBRARYL display. Members are purged by entering the P command on a LIBRARY display. Messages are purged by entering the P command on a LIBRARYQ display.

On VSE, POWER job entries are purged by entering the P command on a LIBRARYP display and VSE sublibrary members are purged by entering the P command on a LIBRARYD display.

On MVS, JES jobs or data sets are purged by entering the P command on a LIBRARYP display and PDS members are purged by entering the P command on a LIBRARYD display.

Example

Purge members starting with "DFHTCT":

```
=>
DISP  -> LIB DFH.                                SESS=A 3( 3) LINE=   79(   95)
-----1-----2-----3-----4-----5-----6-----7--
*==== DFHSIT7T      JCL      CICS 1.7 SYSTEM INITIALIZATION TABLE
*==== DFHSNT7P      JCL      CICS 1.7 SIGN ON TABLE
*==== DFHSNT7T      JCL      CICS 1.7 SIGN ON TABLE *TEST*
*==== DFHSRPS1      JCL      OLTD 1.3 DFHSRP STAGE 1
*==== DFHSRPS2      JCL      OLTD 1.3 DFHSRP STAGE 2
pp==== DFHTCTCB      JCL
*==== DFHTCT6P      JCL
*==== DFHTCT6T      JCL
*==== DFHTCT7P      JCL      CICS 1.7 TERMINAL CONTROL TABLE
*pp==== DFHTCT7T      JCL      CICS 1.7 TERMINAL CONTROL TABLE
*==== TCTCB         ASM
*==== TCTCNSL       ASM
*==== TCTDA         JCL
*==== TCTIRCPD      ASM
*==== TCTIRCTS      COBOL
*==== TCTLL         JCL
*==== TCTTS         ASM
-- END OF DISPLAY --
```

Q(n),QQ

Query / alter a library, member, POWER job entry or JES job or datasets.

Libraries are queried / altered by entering the Q command on a LIBRARYL display.
Members are queried / altered by entering the Q command on a LIBRARY display.

On VSE, POWER job entries are queried / altered by entering the Q command on a LIBRARYP display.

On MVS, JES jobs or datasets are queried / altered by entering the Q command on a LIBRARYP display.

In all cases, the Q command creates a display of the entity. Attributes of the entity can then be updated by overtyping the existing attributes if you have authority to ALTER the entity.

Example

Query / alter members BICADDX, BICALTM, and BICALTP:

```
=>
DISP -> library bic,det                      SESS=A 1( 1) LINE=      0( 174)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----
*====* -- TOP OF DISPLAY --
*====* LIBRARY=ED51A
*====* -----
```

	MEMBER	TYPE	ATTR	---	TEXT	AUDIT	---	DATE	---
				---			---		---
				---			---		---
q====*	BICADDX	ASM	NL	49	0	03/05/1995			
====	BICALTL	ASM	NL	71	0	03/05/1995			
q2====*	BICALTM	ASM	NL	112	114	03/05/1995	06/05/1995		
====	BICALTP	ASM	NL	162	518	03/05/1995	06/23/1995		
====	BICALTS	ASM	NL	70	8	03/05/1995	03/28/1995		
====	BICALTU	ASM	NL	117	215	03/05/1995	05/25/1995		
====	BICATTA	ASM	NL	66	51	03/05/1995	06/12/1995		
====	BICATTX	ASM	NL	68	0	03/05/1995			
====	BICAUCL	ASM	NL	99	0	03/05/1995			
====	BICAUDF	ASM	NL	39	0	03/05/1995			
====	BICAUDI	ASM	NL	42	0	03/05/1995			
====	BICAURL	ASM	NL	242	191	03/05/1995	05/29/1995		
====	BICAURP	ASM	NL	112	64	03/05/1995	05/18/1995		
====	BICAUSM	ASM	NL	222	17	03/05/1995	05/18/1995		
====	BICAUTH	ASM	NL	51	0	06/16/1995	06/16/1995		

R(n),RR

Release a POWER job entry or a JES job or data set.

A POWER job entry or JES job or data set is released by entering the R command on a LIBRARYP display.

VSE Example

On VSE, release a group of POWER LST jobs:

```
=>
DISP -> libraryp lst                      SESS=A 1( 1) LINE= 0( 17)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----
*====* -- TOP OF DISPLAY --
*====*
*====* ----- POWER JOB ----- P D C S   CARDS/ CPY FORM   ORIGIN/   USER INFO
*====* QUE      NAME    NUMBER SG          PAGES          DEST
*====* -----
rr==* LST BIMWNC    11195    3 H A      150    1          BDB
*====* LST BIMWNX    11197    3 H A      22    1          BDB
*====* LST CICDEF    11890    3 H A      15    1          CIC2
*====* LST AMA310OC  11993    3 H A      61    1          CIC2
rr==* LST VSDFSND   12144    3 H A       9    1          JIMF
*====*
*====* -- END OF DISPLAY --
```

VSE Example

On MVS, release a group of JES jobs and data sets:

```
=>
DISP -> libraryp                      SESS=A 1( 1) LINE= 0( 17)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----
*====* -- TOP OF DISPLAY --
*====*
*====* ----- INPUT QUEUE -----
*====* JOB ---- T C RMT PR POS EXEC STAT
*====* NAME  NUMBER Y L NODE TY      SYS  -US
*====* -----
*====* ----- HELD QUEUE -----
*====* JOB ---- T C RMT FORM FCB   CREATE   LINES  ROOM   PROGRAMMER
*====* NAME  NUMBER Y L NODE  ID      DATE      NMBR
*====* -----
rr==* BIM001B  8075 J X    1 STD  ****  08/11/1996    90
rr==* BICEDTD  8083 J X    1 STD  ****  08/11/1996   347
*====*
*====* ----- OUTPUT QUEUE -----
*====* JOB ---- T C RMT FORM FCB   CREATE   LINES  PROC- PRI STAT
*====* NAME  NUMBER Y L NODE  ID      DATE      ESSED  -US
*====* -----
*====* -- END OF DISPLAY --
```


S(n),SS

Process a member based upon its member type.

The LCA S command invokes the PROCESS command. PROCESS processes the member by invoking the SUBMIT, COMPILE, or EXECUTE command depending upon the member type. The following table shows the command invoked for each member type:

Type	Command	Type	Command
ASM	COMPILE	PLI	COMPILE
COBOL	COMPILE	PROC	EXECUTE
FORT	COMPILE	RPG	COMPILE
JCL	SUBMIT		

The function of PROCESS is often customized to support other member types.

Example

Submit member BIJDCUPD for batch processing:

```
=>
DISP -> library bij                      SESS=A 1( 1) LINE=    0( 37)
-----1-----2-----3-----4-----5-----6-----7--
*==== -- TOP OF DISPLAY --
*==== LIBRARY=ED51A
*====
*==== MEMBER          TYPE          TITLE
*====-----
*==== BIJDCLSR        JCL          PRINT MANUAL TO LASER PRINTER
*==== BIJDCCSYS        JCL          PRINT MANUAL TO SYSTEM PRINTER
s==== BIJDCUPD        JCL          TRANSFER BIZUSRF TO TEST LIBRARY
*==== BIJDEF          JCL          DEFINE VSE LIBRARY
```

T(n),Π

Create a DISP session of a member, a DISPI session of an ICCF member or a DISPD session of a VSE/ESA sublibrary member or MVS PDS member.

DISP sessions are created by entering the T command on a LIBRARY display. DISPI sessions are created by entering the T command on a LIBRARYI display. DISPD sessions are created by entering the T command on a LIBRARYD display.

Example

Display member \$SYS.UDP.BIMISCAN

```
=>
DISP -> library $sys.udp.,det                      SESS=A 1( 1) LINE=      0(   52)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==== -- TOP OF DISPLAY --
*==== LIBRARY=$SYS.UDP
*====
*==== MEMBER          TYPE      ATTR      --- LINES --- DATE -----
*==== TEXT  AUDIT      CREATE      UPDATE
*====-----
*==== ALLMEMS          PROC              63      0 06/09/1991 06/09/1991
*==== BIMCICS          COBOL      CL       31      0 04/12/1990 04/12/1990
t==== BIMISCAN          JCL              22      0 12/05/1991 12/06/1991
*==== BIMPRNT          DATA         504      0 08/09/1989 05/23/1990
*==== BULKHELP          TEXT          16      0 11/18/1991 11/18/1991
```

Resulting session

```
=>
DISP $SYS.UDP.BIMISCAN                      SESS=A 2( 2) LINE=      0(   22)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==== -- TOP OF MEMBER --
*==== * $$ JOB JNM=ICCFSCAN,CLASS=0,DISP=D
*==== * $$ LST DISP=H,CLASS=Q
*==== // JOB ICCFSCAN
*==== // TLBL MTOUT,'ICCF LIB BACKUP'
*==== // ASSGN SYS010,182          ICCF LIBRARY BACKUP TAPE
*==== * FORMAT:  PUND=PROG1234      PUNCH PROGRAM
*==== * FORMAT:  LIB=NNN            LIBRARY TO BE SEARCHED OR ALL
*==== * FORMAT:  SCAN=NN,'ARG'      SCAN FORMAT
```

U(n),UU

Translate text to upper case for a specified number of lines.

If the U form is used, n specifies the number of lines to update. If n is not specified, one line is updated. If the UU form is used, UU must be entered on both the starting and ending line of the group of lines to be translated. The upper case translation operates with the zone for the current session.

Example

EDIT session before U command

```
=>
EDIT  NA32.CUSTLIST                      SESS=A 1( 1) LINE=   54(   60)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*
u4====* reliable heat & air
*====* 498 stevenson
*====* p.o. box 103
*====* st. peter, minnesota 54389
*====*
*====* -- END OF MEMBER --
```

EDIT session after U command

```
=>
EDIT  NA32.CUSTLIST                      SESS=A 1( 1) LINE=   54(   60)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*
*====* RELIABLE HEAT & AIR
*====* 498 STEVENSON
*====* P.O. BOX 103
*====* ST. PETER, MINNESOTA 54389
*====*
*====* -- END OF MEMBER --
```

W(n),WW

Translate text to lower case for a specified number of lines.

If the W form is used, n specifies the number of lines to update. If n is not specified, one line is updated. If the WW form is used, WW must be entered on both the starting and ending line of the group of lines to be translated.

Example**EDIT session before W command**

```
=>
EDIT  NA32.CUSTLIST                      SESS=A 1( 1) LINE=   54(   60)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*
w4====*  RELIABLE HEAT & AIR
*====*    498 STEVENSON
*====*    P.O. BOX 103
*====*    ST. PETER, MINNESOTA  54389
*====*
*====*  -- END OF MEMBER --
```

EDIT session after W command

```
=>
EDIT  NA32.CUSTLIST                      SESS=A 1( 1) LINE=   54(   60)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*
*====*  reliable heat & air
*====*    498 stevenson
*====*    p.o. box 103
*====*    st. peter, minnesota  54389
*====*
*====*  -- END OF MEMBER --
```

X(n),XX

Create a INQUIRQP session display of JES or POWER datasets within a job.

INQUIREP sessions are created by entering the X command on a LIBRARYP display. For VSE, this is only valid on a currently executing RDR queue entry.

VSE Example

INQUIREP for job BIMTMAN

```
=>
DISP -> dq rdr,free                      SESS=A 1( 2) LINE= 0( 18)
-----1-----2-----3-----4-----5-----6-----7--
***** -- TOP OF DISPLAY --
*****
***** POWER JOB ----- P D C S CARDS/ CPY FORM ORIGIN/ USER IFO
***** QUE NAME NUMBER SG ----- PAGES ----- DEST -----
*****
***** RDR VTAM 6834 3 * 3 10 BM01
***** RDR CICSProd 6842 3 * 4 48 BM01
***** RDR BIJEDITG 9475 3 * 5 12 5.6A 99 SSSSIONS
***** RDR BIMWNDOW 6838 3 * 8 7 BM01-BW45
***** RDR BIMEDIT 9214 4 * 9 8 EDPD + ED4AD
***** RDR EDITBACK 9696 3 * C 11 BIM
***** RDR BIMTMAN 6839 3 * G 76 BIMTMAN . POD . JCL
***** RDR JCLSCHEd 6843 3 * G 6
***** RDR FAQXCONS 6836 3 * H 5
***** RDR BIJEDITT 9580 6 * J 24 T+P+0AD+EPD+54A
***** RDR CICSTEST 8738 3 * T 170
***** RDR FAQS 6837 3 * V 5
***** RDR BIMTCPT 6841 3 * V 12
***** RDR BIMTCPIP 6840 3 * Y 12
*****
***** -- END OF DISPLAY --
```

Resulting session

```
=>
DISP -> INQUIREP ,,BIMTMAN 6839          SESS=A 2( 2) LINE= 0( 7)
-----1-----2-----3-----4-----5-----6-----7--
***** -- TOP OF DISPLAY --
***** POWER ACTIVE QUEUE DISPLAY
*****
***** --POWER JOB -- C CARDS/ DESTINATI
***** PP QUE, CUU NAME NUMBER L PAGES FORM USER INFO NODE U
*****
***** G1 LST,02E BIMTMAN 6839 Q 210 BIMTMAN.PROD.JCL BIM R00
***** G1 PUN,02D BIMTMAN 6839 Q 0 BIMTMAN.PROD.JCL BIM R00
*****
***** -- END OF DISPLAY --
```

MVS Example

INQUIREP for job PG\$\$\$XIT

```
=>
DISP -> dq o                      SESS=A 1( 1) LINE= 0( 12)
-----1-----2-----3-----4-----5-----6-----7--
***** -- TOP OF DISPLAY --
***** OUTPUT QUEUE -----
***** JOB ----- T C RMT FORM FCB CREATE LINES PROC- PRI STAT
***** NAME NUMBER Y L NODE ID DATE DATE ESSED -US
*****
***** PG$$$PWR 9724 J A 0 STD **** 04/25/1997 52 0 9
***** PG$$$ZAT 9725 J A 0 STD **** 04/25/1997 52 0 9
***** PG$$$XIT 9727 J A 0 STD **** 04/25/1997 52 0 9
***** PG$$$IES 9726 J A 0 STD **** 04/25/1997 417 0 9
***** PG$$$DIR 9728 J A 0 STD **** 04/25/1997 244 0 9
```

Resulting session

```
=>
DISP  -> INQUIREP 9727,1.1.1          SESS=A 2( 2) LINE=    0(   15)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7-----
*====* -- TOP OF MEMBER --
*====* PG$$$XIT/9727 DATASETS
*====* -----
*====* DSET   STEP      PROC      DDNAME  LINES  C
*====* NMBR              STEP              L
*====* -----
*====*    1                      JESJCLIN      14
*====*    2 JES2                      JESMSGLG     16 A
*====*    3 JES2                      JESJCL      12 A
*====*    4 JES2                      JESYSMSG     19 A
*====*    5 JES2                      $INTTEXT      0
*====*    6                      $JOURNAL      0
*====*   101 BIMUTIL                  SYSIN         2
*====*   102 BIMUTIL                  SYSPRINT      5 A
*====*   103                      SYSUDUMP      0 A
*====*   104 BIMUTIL                  SYSPUNCH     58 B
```

Y

Attach to library from the LIBRARYL display.

Example

Attach to library \$SIT.MODEL:

```
=> lib
DISP  -> libraryl                                SESS=A 1 ( 1) LINE=      0 ( 47)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF DISPLAY --
*====* -----
*====* LIBRARY                                TITLE                                USER      SEC
*====* -----
*====* $SIT.COMP                            SITE COMPILE PROCS                            $SYS      DEFS
*====* $SIT.CTRL                            SITE CONTROL LIBRARY                            $SYS      DEFS
*====* $SIT.HELP                            SITE HELP SCREENS                             $SYS      DEFS
y====* $SIT.MODEL                            SITE MODELS                                    $SYS      DEFS
*====* $SIT.PROC                            SITE PROCEDURES                                $SYS      DEFS
```

(Note the LIB command on the command line.)

Resulting Session

```
=>
DISP  -> lib $SIT.MODEL                          SESS=A 2 ( 2) LINE=      0 ( 20)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====* -- TOP OF DISPLAY --
*====* LIBRARY=$SYS.MODEL
*====* -----
*====* MEMBER                                TYPE                                TITLE
*====* -----
*====* $DFL                                COBOL
*====* A                                  ASM
*====* ASM                                ASM
*====* B                                  B
*====* C                                  COBOL
*====* COB                                COBOL
*====* COBOL                              COBOL
*====* DATA                              DATA
*====* E                                  E
*====* FORT                                FORT
*====* ICCF                                ICCF
```

"(n),"

"(n),"

The "(n) is used to duplicate a line n times. If n is not specified, the line will be duplicated once.

The ""(n) is used to duplicate a group of lines n times. The (n) must be on the first "" in the set. If n is not specified, the group of lines will be duplicated once.

When BIM-EDIT responds, the cursor will be positioned at the first copy of the duplicated line.

This command does not use \$STACK.

Example

EDIT session before " command

```
=>
EDIT 4521.RXMA325                      SESS=A 1( 1) LINE= 1( 2)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*      DCL
"4====*      CF INV BAL                  PIC 'ZZZZZZ9V.99',
-- END OF MEMBER --
```

EDIT session after " command

```
=>
EDIT 4521.RXMA325                      SESS=A 1( 1) LINE= 1( 6)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*      DCL
*====*      CF INV BAL                  PIC 'ZZZZZZ9V.99',
*====*      CF INV BAL                  PIC 'ZZZZZZ9V.99',
*====*      CF INV BAL                  PIC 'ZZZZZZ9V.99',
*====*      CF INV BAL                  PIC 'ZZZZZZ9V.99',
*====*      CF INV BAL                  PIC 'ZZZZZZ9V.99',
-- END OF MEMBER --
```


Example

The line which has the backslash in the LCA will become the last line on the screen.

EDIT session before \ command

```
=>
EDIT  ES34.ESGL300                                SESS=A 1( 1) LINE=   21(   234)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*      IF TXM_EOF THEN
*====*      DO;
*====*          PUT SKIP EDIT
*====*              ('** COMPANY "') (A)
*====*              (C COMPANY)      (A)
*====*      \====*              ('" DOES NOT EXIST **') (A);
*====*      END;
*====*      TXM RC  = 'M';
*====*      TXM MBR = 'EXTA';
```

EDIT session after \ command

```
=>
EDIT  ES34.ESGL300                                SESS=A 1( 1) LINE=   24(   234)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*      GET EDIT (CARD) (A(80));
*====*      TXC COMPANY = C COMPANY;
*====*      READ FILE (TXMFL) INTO (TXC_XRCD) KEY (TXC_KEY);
*====*      IF TXM_EOF THEN
*====*      DO;
*====*          PUT SKIP EDIT
*====*              ('** COMPANY "') (A)
*====*              (C COMPANY)      (A)
*====*      \====*              ('" DOES NOT EXIST **') (A);
```

/

/

The line which has the slash in the LCA will become the current line.

Example

EDIT session before / command

```
=>
EDIT  ES34.ESGL300                                SESS=A 1( 1) LINE=   21(   234)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*      GET EDIT (CARD) (A(80));
*====*      TXC COMPANY = C COMPANY;
*====*      READ FILE (TXMFL) INTO (TXC XRCD) KEY (TXC KEY);
*====*      /====*      IF TXM EOF THEN
*====*          DO;
*====*              PUT SKIP EDIT
*====*                  (** COMPANY '') (A)
*====*                  (C COMPANY) (A)
*====*                  (' DOES NOT EXIST **') (A);
```

EDIT session after / command

```
=>
EDIT  ES34.ESGL300                                SESS=A 1( 1) LINE=   24(   234)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*      IF TXM EOF THEN
*====*          DO;
*====*              PUT SKIP EDIT
*====*                  (** COMPANY '') (A)
*====*                  (C COMPANY) (A)
*====*                  (' DOES NOT EXIST **') (A);
*====*      END;
*====*      TXM RC  = 'M';
*====*      TXM_MBR = 'EXTA';
```

<(n),<<(n)

Shift a specified number of lines n columns to the left.

If the < form is used, only one line is updated. If the << form is used, << must be entered on both the starting and ending line of the group of lines to be shifted.

Note that the < form is different from other LCA commands in that n specifies the number of columns to shift rather than the number of lines that are to be updated.

Regardless of the form used, n specifies the number of columns to shift. If n is not specified, the line will be shifted left one column.

Example

EDIT session before <,<< commands

```
=>
EDIT 4216.CFACT                                SESS=A 1( 1) LINE= 17( 234)
----|----1----|----2----|----3----|----4----|----5----|----6----|----7--
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
<6==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
<<2=* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
<<==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
```

EDIT session after <,<< commands

```
=>
EDIT 4216.CFACT                                SESS=A 1( 1) LINE= 17( 234)
----|----1----|----2----|----3----|----4----|----5----|----6----|----7--
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* E QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* E THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* E THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* E THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
```

>(n),>>(n)

Shift a specified number of lines n columns to the right.

If the > form is used, only one line is updated. If the >> form is used, >> must be entered on both the starting and ending line of the group of lines to be shifted.

Note that the > form is different from other LCA commands in that n specifies the number of columns to shift rather than the number of lines that are to be updated.

Regardless of the form used, n specifies the number of columns to shift. If n is not specified, the line will be shifted right one column.

Example

EDIT session before >,>> commands

```
=>
EDIT  4216.CFACT                                SESS=A 1( 1) LINE=   17(   234)
-----1-----2-----3-----4-----5-----6-----7--
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
>6==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
>>2=* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
>>==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
```

EDIT session after >,>> commands

```
=>
EDIT  4216.CFACT                                SESS=A 1( 1) LINE=   17(   234)
-----1-----2-----3-----4-----5-----6-----7--
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==*      SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==* SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==*      SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==*      SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==*      SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==*      SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
*==*      SEE THE QUICK BROWN FOX JUMP OVER THE LAZY BROWN DOG
```

+(n),++

Center text within the session zone for a specified number of lines.

If the + form is used, n specifies the number of lines to update. If n is not specified, one line is updated. If the ++ form is used, ++ must be entered on both the starting and ending line of the group of lines to be centered.

Example

The following example assumes the session zone is 1-72:

EDIT session before ++ commands

```
=>
EDIT  4216.MANUAL                      SESS=A 1( 1) LINE=   17( 4238)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*
*====* ACCOUNTS RECEIVABLE SYSTEM
*====*
*====* User Reference Manual
*====* Release 1.72
*====* ++====* October 1, 1995
*====*
*====*
```

EDIT session after ++ commands

```
=>
EDIT  4216.MANUAL                      SESS=A 1( 1) LINE=   17( 4238)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*
*====*                                ACCOUNTS RECEIVABLE SYSTEM
*====*
*====*                                User Reference Manual
*====*                                Release 1.72
*====*                                October 1, 1995
*====*
*====*
```

(n,((

Left-justify text within the session zone for a specified number of lines.

If the (form is used, n specifies the number of lines to update. If n is not specified, one line is updated. If the ((form is used, ((must be entered on both the starting and ending line of the group of lines to justify.

Example

The following example assumes the session zone is 1-72:

EDIT session before ((commands

```
=>
EDIT  4216.MANUAL                      SESS=A 1( 1) LINE=   17( 4238)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*
*(====*                                ACCOUNTS RECEIVABLE SYSTEM
*====*
*====*                                User Reference Manual
*====*                                Release 1.72
*(====*                                October 1, 1995
*====*
*====*
```

EDIT session after ((commands

```
=>
EDIT  4216.MANUAL                      SESS=A 1( 1) LINE=   17( 4238)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*
*====* ACCOUNTS RECEIVABLE SYSTEM
*====*
*====* User Reference Manual
*====* Release 1.72
*====* October 1, 1995
*====*
*====*
```

)n,))

Right-justify text within the session zone for a specified number of lines.

If the) form is used, n specifies the number of lines to update. If n is not specified, one line is updated. If the)) form is used,)) must be entered on both the starting and ending line of the group of lines to justify.

Example

The following example assumes the session zone is 1-72:

EDIT session before)) commands

```
=>
EDIT  4216.MANUAL                      SESS=A 1( 1) LINE=   17( 4238)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*
*)====*                                ACCOUNTS RECEIVABLE SYSTEM
*====*
*====*                                User Reference Manual
*====*                                Release 1.72
*)====*                                October 1, 1995
*====*
*====*
```

EDIT session after)) commands

```
=>
EDIT  4216.MANUAL                      SESS=A 1( 1) LINE=   17( 4238)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====*
*====*                                ACCOUNTS RECEIVABLE SYSTEM
*====*
*====*                                User Reference Manual
*====*                                Release 1.72
*====*                                October 1, 1995
*====*
*====*
```

Chapter 6. Advanced Techniques

This chapter discusses techniques that can be useful after you become comfortable with BIM-EDIT. These techniques make it faster to get work done with BIM-EDIT or exercise specific features of the product. They are:

- Extended Search Patterns
- Multiple Operations on a Screen
- Using Sequences of Commands
- Split-Screen Operation
- Logical Tabbing
- Hexadecimal Display and Editing
- Member Stamping
- Member Auditing
- Checkout/Checkin Facility

Two other topics which might be thought of in the "Advanced Techniques" category are covered in the BIM-EDIT System Reference Manual:

- Batch Utility
- Writing BIM-EDIT procedures

Extended Search Patterns

Sometimes you need to search for something more complex than a string of characters. For example:

- text that can have several variant spellings
- text only when it occurs on a line WITHOUT other specified text
- text only in the form of a complete word
- text patterns in which some of the characters may vary

BIM-EDIT Extended Search Patterns allow you to make these and other types of complex searches.

Extended Search Pattern Characters

Extended Search Patterns assign special meanings to the characters shown below. These meanings only take effect if the search string begins with a backslash (\) character -- otherwise every character in a search string represents itself. (There is one exception to the preceding sentence for the entity name operand of commands that operate on groups of BIM-EDIT entities: LIBRARY, LIBRARYL, LIBRARYU, and SCAN. These commands accept ? and * in the entity name and do not accept a preceding backslash on the entity name.)

In an Extended Search Pattern:

- ? (single character wildcard) matches any single character. '\A?M' matches 'ABM' or 'ASM'.
- * (many character wildcard) matches any sequence of zero or more characters. '\I A*M' matches 'I AM' or 'I ARM' or 'I ANTIDISESTABLISHMENTARIANISM'.
- @ (repeated character wildcard) matches zero or more instances of the character that follows it; also suppresses any special function of that character. '\X@-Y' matches 'XY', 'X-Y', or 'X-----Y'.
- < (begin word) matches the start of the search area or any non-alphanumeric character. '\<HAT>' finds a match in 'THE OLD FELT HAT', but does not find one in 'THAT MAD HATTER'.
- > (end word) matches the end of the search area or any non-alphanumeric character. See example for > above.
- ! (begin area) matches the start of the search area. '\!XAS' will not match 'TEXAS'
- + (AND) combines two sub-patterns BOTH of which must match. '\SAINT+LOUIS' matches 'LOUIS, THE PATRON SAINT OF PROGRAMMERS' but does not match 'ST. LOUIS' or 'SAINT LOUIE'.
- | (OR) combines two sub-patterns EITHER of which may match. '\<SC>|<TN>' matches both 'GREENVILLE, SC' and 'KNOXVILLE, TN'.

- ~ reverses the match failure/success of the sub-pattern which follows. '\~JONES' will match any search area that does not contain 'JONES'.
- \ (suppress) indicates the character which follows it is NOT to be treated as an Extended Search Pattern character. '\X*\+ Y' matches 'X + Y' and 'XANADU, JOHNSON + YOLANDA'. Without the \, it would have matched any search area that contained both 'X' and 'Y', taking into account the special meaning of +.
- % (position marker) indicates the position in the search area at which the match is considered to have taken place. This is relevant for the CHANGE command and the CURSOR command. For the CHANGE command, a second % may be used to indicate the end of the portion which is to be replaced. It can also be used within a procedure to set the variables PPD COL, SSD COL1, and SSD COL2.

The meaning of "search area" in the preceding descriptions varies according to context. In CHANGE, LOCATE, LOCATEU, QUALIFY, and the STR operand of the SCAN command, "search area" means the ZONE column range of a session line. In FIND, FINDUP, NFIND, and NFINDUP, "search area" means the columns from COL to the end of a session line. In EXAMINE, LIBRARY, LIBRARYL, LIBRARYU, SCAN (other than the STR operand), and VEXAMINE, "search area" means the value of the variable or attribute referenced.

Using Extended Search Pattern to Search for "Words"

Sometimes, it is useful to find instances of a particular word but not find instances that are contained in other words. The < and > characters can be used in an Extended Search Pattern to represent a non-alphanumeric character which begins or ends a word, or the start or end of the search area.

For example, to position to the next instance of the word 'we', but not words that contain it like 'were' or 'tower', you could enter

```
=> locate \<we>
```

To display all instances of variable names in a program which start with 'N' but not any of the other variable names that contain 'N', you could enter

```
=> qualify \<N
```

Because they function by matching non-alphanumeric characters and then moving the search pointer forward or backward, < and > may appear to behave oddly if they are not at the beginning and end of the search string. For example, the search string '\ABC<DE>F' will never match anything and the search string '\>#<' will match a single # anywhere.

There are several ways to match multiple words, depending on what function you need:

```
=> locate '\<major>+<standards>' both words on same line, in any order
=> locate '\<major standards>' consecutive, one blank between
=> locate '\<major>*<standards>' in order on line, any characters between
=> locate '\<major@ standards>' consecutive, any number of blanks between
```

Of course, for a multiple word match to succeed, the words must all appear on a single line, not be split across lines.

Using Extended Search Pattern "Wildcards"

Extended Search Patterns provide several ways to indicate unknown characters. ? represents one unknown character, * represents zero or more unknown characters, and @ indicates zero or more repeats of the character which follows it. These can be combined for various effects:

'\A?*B' 'A' separated from 'B' by at least one character.

'\A????*B' 'A' separated from 'B' by at least four characters.

'\A\!@!B' 'A' separated from 'B' by one or more ! characters. ('\A@\!B' will never match anything because @! consumes all repeats of !)

Using the * at the beginning of a string will make a fixed column search command operate like a variable column one. For example,

```
=> find \*metamagical col=8
```

will find the next instance of the word "metamagical" between columns 8 and 253, operating essentially like a LOCATE command.

Extended Search Pattern Combinations

Sometimes, it is useful to search for several different spellings of a name or find instances of a string which appear with (or do not appear with) another text string. Extended Search Patterns provide the + (AND), | (OR), and ~ (NOT) for these situations.

For example, to find the next instance of any of three alternate spellings for Kelsey, you could enter

```
=> locate '\<Kelsey>|<Kelsy>|<Kelsie>'
```

To display all lines in which Larson appears without Peterson, you could enter

```
=> qualify '\<Larson>+~<Peterson>'
```

+ and | function by dividing the search string into sub-patterns. Sub-patterns are matched in strictly left-to-right sequence until the search string is fully consumed or the outcome is judged to be determined. The outcome is judged to be determined when the character following a sub-pattern is | and the sub-pattern matches or when the character following a sub-pattern is + and the sub-pattern does not match.

~ functions by reversing the match results of the sub-pattern which follows it.

When you mix + and | in an Extended Search Pattern, you should give the most controlling sub-pattern first. For example, if you want to search for the next line where Flaws appears with Kingsbury or Maloney, you would enter

```
=> locate '\<Flaws>+<Kingsbury>|<Maloney>'
```

If you entered

```
=> locate \<Kingsbury>|<Maloney>+<Flaws>
```

the result would be the next line where Kingsbury appeared or the next line where Maloney appeared with Flaws.

Not all Boolean conditions can be represented with these rules. For example, there is no way to search for "Larson without Peterson or Peterson without Larson".

The use of + combinations and ~ are not recommended with the CHANGE command because the start and end positions for replacement may not be what you would expect.

Making Search Commands Work Like One Another

For example, to find the next line in the current session that does NOT contain the string MALLARD at column 10, you could enter any of the following:

```
=> nfind MALLARD 10
=> find \~MALLARD 10
=> locate \~!MALLARD zone=10-*
```

Using Extended Search Patterns in this way may be confusing if you inherit STR values from command to command.

Using the % Extended Search Pattern Position Marker

Sometimes it is useful if the column position at which matching is considered to take place is other than the first character of the search string. Extended Search Patterns provide % for this purpose. % indicates the column at which the CURSOR command will place the cursor and the column at which the CHANGE command will insert the replacement string. If the pattern does not contain a %, the match is considered to take place at the first non-pattern character which participated in the match.

For example, to position to the next line which contains '// EXEC ' and position the cursor after it, you could enter

```
=> locate '\\// EXEC %';cursor
```

You can use % with + and | by following the rule that % should appear in ALL sub-patterns combined with | but only in ONE of the sub-patterns combined with +. For example, to position to the next line which contains the string EXTENT and the string VOL123 and place the cursor at VOL123, you could enter

```
=> locate \EXTENT+%VOL123;cursor
```

% is ignored when used with ~.

In the CHANGE command, a second % may appear in to mark the end of the characters to be removed. If there is no second %, the last non-pattern character participating in the match will be the last character replaced.

For example, to replace all instances of "ITM-PRICE + ITM-COMM" with "ITM-PRICE - ITM-COMM", you could enter

```
=> change '\ITM-PRICE %\+% ITM-COMM' '-' *
```

When the position indicated by % does not follow or immediately precede a non-pattern character, the position set by % may not be what you expect. For example, the pattern '%?ADAM' will set the position to the beginning of search area containing "ADAM", not to the character preceding "ADAM". This situation occurs when % is mixed with wildcard indicators at the beginning of an Extended Search Pattern. When using the CHANGE command with % and + or |, beginning and ending % should both be in the same sub-pattern. Otherwise the start and end positions for replacement may not be what you would expect.

Finding First Non-Blank in a text Line

A common use of the extended search string is to position the cursor at the first non-blank character in a line. The following locate command will perform this function:

```
=> locate "@ %?";cursor
```

A common mistake is to use the following locate command:

```
=> locate "~ ";cursor
```

This search will only position to text lines that do not contain any blanks because the search request says, in effect, find a blank anywhere in the search area, and then reverse the match to a false condition.

Multiple Operations on a Screen

As you get more proficient with BIM-EDIT, you will find that it is generally faster to perform several operations on each screen. The degree to which this speeds up editing will depend on your computer's response time -- if you use BIM-EDIT over low speed dial-up communication to a heavily loaded computer you will save much more time by performing several operations on one screen than if you are directly connected to a lightly loaded computer.

The procedure of experienced BIM-EDIT users on each screen is:

1. Make any changes in the text area.
2. Use the LCA commands to copy, move, delete, or add any lines.
3. Use the command line to perform any other reformatting
4. Use the LCA, command line or a PF key to reposition to the next location to view or update.

The order of processing operations on the screen determines the net effect of multiple operations. (BIM-EDIT receives the screen as a block, with no information on the order in which you made your entries.)

BIM-EDIT's default order of processing is:

1. The text display area. Updateable portions of the text display area(s) are scanned from top to bottom of the screen for changed lines and are modified. (If you are in split screen mode and have the same text lines showing on both logical screens, edits made on the lower logical screen will take precedence over conflicting edits made on the upper logical screen.)
2. The LCA area.
 - a. The LCA is scanned from top to bottom of the screen to process any commands that add lines to \$STACK.
 - b. The LCA is again scanned from top to bottom to process any other commands.
3. The command line. Command(s) are processed in a left-to-right order.
4. If a special key (PF1-PF24, PA1-3, CLEAR) was depressed, the command string assigned to the key will be processed.
5. If no data has been entered anywhere on the screen, and the transmission was initiated by the ENTER key, the command string (if any) assigned to the ENTER key will be processed.

The processing order for steps (3) and (4) can be changed for specific PF keys by specifying a PF key processing option other than the default of 'AFTER'. PF key processing options are discussed in the KEYS and SET commands.

Example

Screen before transmission with PF7 (assigned to BACK)

```

=> o 1 ':' 7;o 7 ':' 7;o 32 ':' 7;o 39 ':' 7;o 72 ':' 7
EDIT  IS.ASMCHART                      SESS=A 1( 1) LINE=      1(    20)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
c====+-----+-----+-----+-----+-----+-----+-----+
*/====Op                               Mne-
*i====Code          Name                monic          Characteristics
=====04      SET PROGRAM MASK          SPM      RR      L
=====05      BRANCH AND LINK           BALR      RR
*i====06      BRANCH ON COUNT            BCTR      RR          B      R
=====07      BRANCH ON CONDITION        BCR       RR          $      B
*i====08      SET STORAGE KEY            SSK       RR      P      A      SP
=====09      INSERT STORAGE KEY         ISK       RR

```

Result of transmission

```

=>
EDIT  IS.ASMCHART                      SESS=A 1( 1) LINE=      0(    23)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*====-- TOP OF MEMBER --
*====+-----+-----+-----+-----+-----+-----+-----+
*====:Op   :                               :Mne-  :
*====:Code :          Name                :monic  :          Characteristics :
*====:-----:-----:-----:-----:-----:-----:-----:
*====:04   :SET PROGRAM MASK              :SPM    :RR      L :
*====:05   :BRANCH AND LINK               :BALR   :RR      :          B      R :
*====:06   :BRANCH ON COUNT              :BCTR   :RR      :          B      R :
*====:-----:-----:-----:-----:-----:-----:-----:
*====:07   :BRANCH ON CONDITION           :BCR    :RR      :          $      B :

```

Using Sequences of Commands

Where many instances of a repetitive change are needed in a member, you can spend a long time performing the change instance-by-instance if it is not of a type that can be done with a global command such as CHANGE or OVERLAY. In many of these situations, it is possible to set up a sequence of commands on the command line which will significantly reduce your labor.

For example, suppose you had decided to change a 10,000 line member so that every instance of the line

```
Example of ...
```

would be changed to

```
-----  
Example of ...  
-----
```

where "..." stands for a varying string of text. To do this without using command sequences, a reasonably optimal procedure would be to use the FIND command to position to each instance, then position up a line, GET a member containing a line of dashes, truncate the dashes to the desired length, copy the truncated dashes after the Example line, center all three lines, and then use the FIND command again. This would be time consuming if there were a hundred instances.

However, you could save work by setting the current line to a line of dashes of the proper length before the first "Example of" line and then entering:

```
=> &cen 2;st 1;n;get $stack;f "Example of" 4;justl;up;get dashes;n;cursor
```

With this sequence, the following will happen every time you hit the ENTER key:

1. the current line (the dashes) and the line following (Example of) will be centered.
2. The current line (the centered dashes) will be copied to \$STACK.
3. Position will be moved forward 1 line (to centered Example of).
4. The line just copied to \$STACK (the centered dashes) will be inserted after the current line (centered Example of).
5. Position will be changed to the next instance of "Example of" in column 4.
6. The current line (Example of) will be justified left.
7. Position will be moved backward 1 line (before Example of).
8. The contents of member DASHES will be inserted (assume a line of dashes.)
9. Position will be moved to the line just inserted (the dashes).
10. The cursor will be positioned on that line.

With this sequence, all that you do to change one instance is to move the cursor to the point on the dashes line above the last character on the "Example of" line, hit the ERASE EOF key to get rid of the extra dashes, and hit the ENTER key.

Sequences are developed by determining the point where manual intervention is required (in this case, truncating the dashes to the proper length) and then entering on the command line all the commands required to perform the other steps). By assigning different sequences to different PF keys, you can even achieve alternative processing.

Of course, setting up a complex sequence to perform an edit which occurs only a few times in a member will take more time than it will save.

(One side-benefit of developing sequences like these is that you practice the kind of commands and the way of thinking you need when you write procedures. Procedures are much more capable than sequences because they can execute commands repeatedly and can execute commands only when specified conditions are met. Procedures are documented in the BIM-EDIT System Reference Manual.)

Split-Screen Operation

In split-screen mode, the screen is divided into two logical screens. This facility can be useful in the following ways.

- The logical screens can be positioned at different points in the same session to compare or copy blocks of text.
- The logical screens can work with different sessions. Two members or a member and a listing can be compared.

To activate split-screen mode, enter:

```
=> screen split,tog
```

The same command can be entered to toggle back to single-screen mode. Refer to the SCREEN command for other options for setting split-screen mode.

If you were editing a member OMRKSIO and activated split-screen mode, a screen of the following format would appear:

```
=>
EDIT 4217.OMRKSIO                                SESS=A 1( 1) LINE= 34( 256)
-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==== MVC TXLBLKSZ,=H'4096' SET CONSTANT TXLFL RECORD SIZE
*==== L R1,PM4 GET BLOCK COUNT ADDRESS
*==== LH R1,0(R1) GET BLOCK COUNT
*==== MH R1,TXLBLKSZ COMPUTE NUMBER OF BYTES NEEDED
*==== STH R1,IOLENGTH SAVE IN I/O CONTROL FIELD
*==== L R1,PM2 GET EXTENT INFO PARM
*==== MVC IOEXTENT,0(R1) USE FILE EXTENT 1 INFO FOR RELOCATE
*==== BAL R11,RELOCATE RELOCATE CHANNEL PROGRAM
*==== BAL R11,CALCSEEK CALCULATE SEEK ADDRESS
~~~~~
EDIT 4217.OMRKSIO                                SESS=A 1( 1) LINE= 0( 256)
< --1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==== -- TOP OF MEMBER --
*==== *****
*==== * SUBROUTINES
*==== * 1) RMZPQAR - HANDLES INPUT FROM CARDS
*==== * 2) RMZPQXR - HANDLES INPUT FROM TAPE
*==== * 3) RMZPQXA - HANDLES INPUT FROM SCREEN
*==== * 4) RMZPQXM - HANDLES INPUT FROM MEMBER
*==== * 5) RMZPRS1 - PARSES INPUT STRING, TYPE 1
*==== * 6) RMZPRS2 - PARSES INPUT STRING, TYPE 2
```

There is only one command line for the two logical screens. Commands act upon the logical screen designated as "primary". The less than sign (<) appears in column 1 of the current primary screen's scale line. Immediately after entering split-screen mode, the lower logical screen is the primary one. (The primary screen concept relates only to commands entered through the command line, not LCA commands, and not updates to the text display area. LCA commands and text updates may be entered for both screens on the same transmission.)

To cause the secondary logical screen to become the primary logical screen, the "<" symbol can be entered as a command. For example, if the lower logical screen is primary, then the command "<top" will cause the upper logical screen to become primary and cause the upper logical screen session to position at the top.

Using "<", you can enter a command string that acts upon both logical screens. For example, if you wanted to position forward a full page for both logical screens, you would enter:

```
=> <;forward;<;forward
```

To work with two different sessions, enter a command to create or ROTATE to another session after the SCREEN SPLIT. If you entered "list omrexop", the following might appear:

```
=>
EDIT  4217.OMRKSIO                                SESS=A 1( 2) LINE=   34( 256)
-----|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==== MVC      TXLBLKSZ,=H'4096'          SET CONSTANT TXLFL RECORD SIZE
*==== L        R1,PM4                      GET BLOCK COUNT ADDRESS
*==== LH       R1,0(R1)                   GET BLOCK COUNT
*==== MH       R1,TXLBLKSZ                COMPUTE NUMBER OF BYTES NEEDED
*==== STH      R1,IOLNGTH                 SAVE IN I/O CONTROL FIELD
*==== L        R1,PM2                      GET EXTENT INFO PARM
*==== MVC      IOEXTENT,0(R1)             USE FILE EXTENT 1 INFO FOR RELOCATE
*==== BAL      R11,RELOCATE                RELOCATE CHANNEL PROGRAM
*==== BAL      R11,CALCSEEK               CALCULATE SEEK ADDRESS
~~~~~
LIST  4217.OMREXOP                                SESS=A 2( 2) LINE=    0( 312)
< --|-----1-----|-----2-----|-----3-----|-----4-----|-----5-----|-----6-----|-----7--
*==== -- TOP OF MEMBER --
*==== * *****
*==== * PROGRAM - OMREXOP
*==== *
*==== * AUTHOR - TIM JOHNSON
*==== *
*==== * DATE - 08/30/78
*==== *
*==== * FUNCTION - PARSE THE INPUT LINE
```

Logical Tabbing

BIM-EDIT provides a logical tabbing facility to align text to columns as you enter it. This is useful when entering text in columns or when entering text with indentation from the left margin (programs are often entered this way).

(IBM provides a 3274 controller feature known as "data entry assist" that provides "physical" tabbing at the controller / screen level. If the controller feature is available, you may find its tabbing facility more useful than the "logical" tabbing facility provided by BIM-EDIT. The entry assist feature also provides a word wrap facility that can be useful for text applications.)

The logical tabbing facility is activated when all of the following are true:

- the TAB character (SIBTABC) has been set to a non-blank value. This is handled by the SET command. For example:

```
=> set sibtabc,@
```

- the current session is an EDIT session.
- the member being edited has tab columns defined. Tab columns are defined with the TCOL operand of the DEFINE, FDEFINE, ALTER, FALTER, or SESS commands. For example:

```
=> sess tcol=8-12-16-20-24-32-40
```

If the tabbing facility is activated, the following occur:

- The SIBTABC character will be displayed on the scale line marking each tab position defined in the sess tcol operand.
- Each new or modified line will be searched for the SIBTABC character.
- Text will be shifted right to the column corresponding to the first tab setting greater than the column where the SIBTABC character occurred.
- If the SIBTABC character appears in the text in a column beyond any logical tab setting, it is ignored for tabbing purposes.
- The SIBTABC character will not appear in the text after the transaction. Any text to the right of the SIBTABC character will be shifted to the left one space.
- The process repeats until all SIBTABC characters have been processed.

Example

SIBTABC has been set to the commercial at sign (@). The member being edited, MMAR040, has tabs for columns 8,12,16,40,56.

Logical tabs in the EDIT session before transmission

```
=>
EDIT  X324.MMAR040                      SESS=A 1( 1) LINE=      8(    20)
-----|--@-1-@--|@---2---|---3---|---@---|---5---|@---6---|---7--
*====* @a10-proc.
*====* @@if switch = 'a'
*====* @@add 1 to counter
*====* @@add 1 to counter2.
*====* @@multiply gross by rate giving commission.
*====* @a20-proc.
*====*
*====*
*====*
```

Result of logical tabbing in EDIT session

```
=>
EDIT  X324.MMAR040                      SESS=A 1( 1) LINE=      8(    20)
-----|--@-1-@--|@---2---|---3---|---@---|---5---|@---6---|---7--
*====*      A10-PROC.
*====*          IF SWITCH = 'A'
*====*              ADD 1 TO COUNTER
*====*              ADD 1 TO COUNTER2.
*====*          MULTIPLY GROSS BY RATE GIVING COMMISSION.
*====*      A20-PROC.
*====*
*====*
*====*
```


Editing Text Hexadecimally

The BIM-EDIT hexadecimal display does not allow lines to be updated by overtyping or using the LCA. Instead, hexadecimal updates occur through the command line. To perform hexadecimal updates:

1. Set the HEX character (SIBHEXC) to a non-blank value. For example:

```
=> set sibhexc,@
```

2. When SIBHEXC has a non-blank value, hexadecimal translation is activated for command line input. (Input coming from sources other than the command line, such as procedures, is not affected by the SIBHEXC setting.)

The command line is scanned for occurrences of the SIBHEXC character. For each occurrence, if the following two characters are valid hexadecimal digits, the three characters (the SIBHEXC value and the two hex digits) are replaced with the one character represented by the two hexadecimal digits.

Usually, the CHANGE or OVERLAY commands are used to perform the edits. For example, the following command would change all slashes (/) to hexadecimal FF characters.

```
=> change /,@ff,*
```

3. To set hex translation off, set SIBHEXC to a blank value as follows:

```
=> set sibhexc,''
```

Hexadecimal Translation at Processing Time

Storing non-displayable characters in members causes difficulties in displaying and editing the lines that contain the characters. The more viable approach provided with BIM-EDIT is to perform the hexadecimal translation when the text is submitted (actually, any "text following" command will perform the same translation -- see the BIM-EDIT System Reference Manual). This technique has the advantage that the text is displayable when you are editing it, yet when it reaches batch processing, the text will be translated.

To activate the processing-time hexadecimal translation facility, predefined variable PPDHEX must be set to "1" and predefined variable PPDHEXC must be set to a non-blank character which is used as the hexadecimal translate character for processing (when processing is initiated, PPDHEX is 0 and PPDHEXC is "@"). Your System Administrator can help you determine how to get the member processed with translation.

For example, if the line

```
DCL SBA CHAR(1) STATIC INIT('@11');
```

is processed with PPDHEX set to 1 and PPDHEXC set to "@", batch processing will receive the single character whose hexadecimal representation is 11 between the quotes in the INIT clause.

Member Stamping

Member stamping saves extra information when a text line is added or updated. The primary benefit of stamping is to be able to see when, where, and by whom changes have been made in stable members. Stamping keeps no information concerning deleted lines.

Stamping may be used together with auditing or in place of auditing. Stamping provides less information than auditing, but the information provided can be displayed more concisely. The resources required for stamping, particularly disk space, are significantly less than those required for auditing.

Use of Stamping

Member stamping is activated in one of the following ways:

- If a new member is created by either the COPY, DEFINE, or FDEFINE commands, and if STAMP=ON was specified for the source/template member, stamping will be active for the new member.
- Stamping can be set on by specifying STAMP=ON in the DEFINE, FDEFINE, ALTER, or FALTER commands.

Member stamping is displayed in the LCA using the STAMP mode of the SCREEN command. Several format options are available.

Member stamping can be printed or replaced into a member using the PPDSPCOL and PPDSPFMT predefined variables with the INSERTF, PRINTF or SUBMITF commands. For more discussion, see the BIM-EDIT System Reference Manual.

Member stamping information can be cleared only through the STAMPCL command.

Format of Stamp Entry

The following information is recorded:

- Type of update - A= new line, M= update of existing line. (A modified line will be update type M only if it was already in the member when stamping was turned on or cleared. A line added to the member will always be type A even if it is subsequently modified. Lines which are moved within the member are deleted and re-inserted, becoming type A in the process.)
- Date of last update.
- User performing last update.

Administration of Stamping

Two administrative decisions should be made concerning stamping:

- 1) Are all members to be stamped? If yes, set stamping on for all members and set MMPSPCTL to "1" to prevent users from setting stamping off for any member. If not, what members are to be stamped? How will this be assured?

- 2) At what point should the stamps be cleared for a member?

Member Auditing

Auditing maintains a chronological line-by-line trace of updates that occur against a member. Auditing provides two benefits:

- If, for any reason, you need to know what updates occurred against a member, you can either view the audit trail directly or create a summary report of changes based upon the audit trail.
- If you need to restore a member to its status prior to certain updates, you can make use of the audit trail to reverse updates. This is handled by the AUDITRL command or by copying lines from the audit trail into the member.

While member auditing can be quite useful, there is a price to be paid in terms of resources consumed.

- There is additional processing time associated with maintaining the audit trail. If a line is updated, up to four lines are written to the audit trail. You probably won't find the additional processing time prohibitive (you may not even notice it), but it is a consideration.
- There is a definite disk space requirement. It is not unusual for a member that has undergone extensive editing to have an audit trail that is four or five times larger than the member text.

The audit trail maintained for a member is unique to that member. The audit trail is linked to the member record in the same way that the member text is linked. The audit trail is thus inseparable from the member. It can be cleared, but only through audit control commands (AUDITCL and ALTER AUDIT operand).

Use of Auditing

Member auditing is activated in one of the following ways:

- If a new member is created by either the COPY, DEFINE, or FDEFINE commands, and if AUDIT=ON was specified for the source/template member, auditing will be active for the new member.
- Auditing can be set on by specifying AUDIT=ON in the DEFINE, FDEFINE, ALTER, or FALTER commands.

A member's audit trail can be displayed by creating an EDIT or LIST session of the member and then entering:

```
=> screen aud,tog
```

(Entering the same command again would switch you back to viewing the member text. If you use auditing, you may want to assign this command to a PF key.)

A member's audit trail can be printed by copying the text of the audit trail into a temporary member (via \$STACK) and then printing the temporary member.

A member's audit trail can be summarized using the AUDITSM command.

Updates against a member can be reversed based upon the audit trail. See the AUDITRL command documentation in Chapter 4, Commands, for instructions. (You can also selectively recover deleted or modified lines by copying the lines from the audit trail into \$STACK and then inserting \$STACK into the member text. You can accomplish this with the LCA commands C, CC, K, KK, I, and B, all of which are documented in Chapter 5, LCA Commands.)

Auditing can be turned off and the audit trail cleared by altering the member's AUDIT attribute to OFF.

Format of the Audit Trail

The audit trail consists of a chronological record of update events. The oldest entries are at the top and the youngest are at the bottom. The audit trail is composed of two types of lines:

- . Text lines are copies of lines from the member. For example, if a line is deleted from the member, a copy of the line is written to the audit trail.

- . Control lines describe the update events. Control lines are often associated with text lines. For example, in the above example of deleting a line, three lines would be written to the audit trail, a .DEL control line, followed by the deleted text line, followed by a second .DEL line.

The following paragraphs describe in detail the events and their associated control lines.

1. When an EDIT session is created, a line of the following format is written:

```
.EDIT  DATE=mm/dd/yyyy, TIME=hh:mm:ss, USER=uuuu, TERM=tttt
```

where

- mm/dd/yyyy and hh:mm:ss are the date and time, respectively, when the EDIT session was created. (Date may be in the dd/mm/yyyy format, depending upon the date format set by your System Administrator.)

mm is the month (01-12)

hh is the hour (00-23)

dd is the day (01-31)

mm is the minute (00-59)

yyyy is the year

ss is the second (00-59)

- uuuu is the user that created the EDIT session.
- tttt is the terminal ID where the user was logged on.

2. When an EDIT session is saved, a line of the following format is written:

```
.SAVE  DATE=mm/dd/yyyy, TIME=hh:mm:ss, USER=uuuu, TERM=tttt
```

The DATE, TIME, USER, and TERM values are interpreted as in the .EDIT control line.

(If a session is ended with the END NOSAVE command, the audit trail will not even show the original .EDIT control line. This is because both the text and the audit trail of an EDIT session are work copies. They become permanent only when the SAVE command is issued.)

3. When updates are reversed via the AUDITRL command, a line of the following format is written:

```
.ROLL  DATE=mm/dd/yyyy,  TIME=hh:mm:ss
```

The DATE and TIME values are interpreted as in the .EDIT control line.

After the .ROLL line, the rollback itself will be audited. For example, if a deletion of 20 lines is rolled back, the audit trail will show the 20 lines inserted. After the rollback is complete, another .ROLL control line is written. This line is the same as the original .ROLL line.

Because the rollback process itself is audited, it is possible to rollback a rollback, if necessary.

4. When comment lines are written to the audit trail by either the AUDITF or AUDITI commands, they have the following format:

```
.**** text
```

The rest of the control lines describe normal updates against the member. All of them have a "LOC=lllll" in their format. The "lllll" is the line number location associated with the update. (Note that line numbers keep changing as updates take place. Line numbers in entries not at the bottom of the audit trail may not match the line numbers currently in the member.) All except the .MOD control line have a "EXT=eeeeee" in their format. The "eeeeee" is the extent associated with the update, that is, the number of lines involved.

5. When blank lines are added, an audit line of the following format is written:

```
.ADD LOC=lllllll, EXT=eeeeee
```

6. When lines are deleted, audit lines of the following format are written:

```
.DEL LOC=lllllll, EXT=eeeeee deleted text lines .DEL LOC=lllllll, EXT=eeeeee
```

7. When lines are inserted, (using a GET or an LCA I command, for example), audit lines of the following format are written:

```
.INS  LOC=lllllll, EXT=eeeeee
inserted text lines
.INS  LOC=lllllll, EXT=eeeeee
```

8. When a line is modified, audit lines of the following format are written:

```
.MOD  LOC=lllllll
text line before update
text line after update
.MOD  LOC=lllllll
```

Administration of Auditing

Two administrative decisions must be made concerning auditing:

- 1) Are all members to be audited? If yes, set auditing on for all members and set MMPAUCTL to "1" to disallow users from setting auditing off for any member. If not, what members are to be audited? How will this be assured?
- 2) At what point should the audit trail be cleared for a member? How much of the audit trail should be cleared? Should the cleared portion be written to tape first? (See the AUDITCL and AUDITRP commands.)

Checkout/Checkin Facility

The checkout/checkin facility is useful to control development and maintenance of members which several users are updating and/or using in common. It was designed to be applied to programming projects, but is as useful for authoring projects and data collection projects.

Checkout/Checkin provides the following benefits:

- It ensures that no two users are working on the same member at any point in time. The checkout/checkin facility requires that a user checkout a member to a working library to edit it. While a member is checked out, the facility ensures that only the user who checked out the member can edit it, and prevents any other user from checking out the member.
- Through the use of the CHECKOUT and CHECKIN commands, it imposes an orderly process of beginning and ending a change. This may provide a juncture to cause ancillary processes such as documentation, testing, and change notification to take place. The change does not become visible to other users until it is checked in.
- Optionally, it can enforce stringent control over how and when an "experimental" version of a member becomes a "production" version. The checkout/checkin facility can be set up so that a user can checkout a member, but only a supervisor can check it in.

Use of Checkout/Checkin

A member is under checkout/checkin control if, and only if, its CHECK attribute is ON (see the CHECK operand of the DEFINE and ALTER commands). If the CHECK attribute is ON, the member text can only be updated through the checkout/checkin mechanism; it cannot be directly edited.

When a member is under checkin/checkout control, you update it as follows:

1. Using CHECKOUT, you cause a working copy of the member to be created and placed in your work library. (The original member is known as the "master" and the working copy is known as the "slave". The master and slave are linked in a way that can only be severed through the CHECKIN or CHECKPUR commands. For example, BIM-EDIT will not allow you to purge the slave or the master through the PURGE command.)
2. You edit, compile, test, etc. the slave.
3. When the member is ready, the CHECKIN command is issued. CHECKIN replaces the master with the slave, then purges the slave. You can use CHECKPUR to simply purge the slave without updating the master.

INCLUDE Processing for Checked Out Members

INCLUDE commands in the checked out slave member are handled somewhat differently than normal. When an INCLUDE is encountered, BIM-EDIT treats the member as if it was part of the master library even though it resides in a user's work library.

Example: Member OM20E.OMREXIO is checked out to library GHM. The following INCLUDE line is imbedded in the member:

```
/INCLUDE OMDURT
```

When member GHM.OMREXIO is submitted for processing, the INCLUDE line will incorporate the text of member OM20E.OMDURT. If GHM.OMREXIO were not a checkout slave, the INCLUDE line would incorporate member GHM.OMDURT. If the user wanted to include member GHM.OMDURT, the INCLUDE line would have to appear as follows:

```
/INCLUDE GHM.OMDURT
```

Administration of Checkout/Checkin

When the checkout/checkin facility is used, the project supervisor should tell your System Administrator how much responsibility to assign to users of the members. Your System Administrator then defines library security accordingly. One of two alternatives is possible:

- Users can check a member out, and can also check it in. In other words, users make the decision themselves as to when a member is ready to be returned to the master. This is accomplished by assigning users EDIT level access to the master library and DEF level access to their work libraries.
- Users can check a member out, but it is checked in by the supervisor. In other words, the supervisor rather than the users makes the decision as to when a member is ready to be returned to the master. This is accomplished by assigning users LIST level access to the master library and DEF level access to their work libraries and assigning the supervisor DEF level access to the master library.

The COPY command may be used to make copies of members which are in a checkout relationship. This is useful because such members may be models used to start a new member or it may be necessary to make a copy in an emergency to resolve a production problem. In the latter case, a manual procedure will be needed to inform the person who has the program "checked out" of the emergency change. In all cases however, use of COPY to obtain an alternate version of a master member which has been checked out to another user should be discouraged.

Checkout/Checkin with Auditing and Stamping

The checkout/checkin facility can be used in conjunction with member auditing and/or member stamping. The STAMP and AUDIT attributes of a slave member cannot be ALTERed. Hence, if auditing and/or stamping is active for the master it is also active for the slave. If the user is not authorized to ALTER members in the master library (access level no higher than EDIT), the supervisor controls whether auditing and/or stamping is on or off. This gives the supervisor a tool to evaluate the updates before making members master.

Further, procedures can be set up to ensure that an audit summary or a stamp listing is produced and kept each time a member is checked in to allow back-tracking if it is necessary to determine when a particular change became production.

Index

The following is a combined index to both the BIM-EDIT User Reference Manual and the BIM-EDIT System Reference Manual. Entries in the User manual are followed with (U); those in the System manual are followed with (S).

' See Apostrophe

\$DFL

 DEFINE Command (U) 100

 FDEFINE Command (U) 139

\$LOG

 Description (U) 24

 to Copy Lines from (GET) (U) 152

 to Display (LIST) (U) 209

 to Insert a Line (LOGI) (U) 226

\$MAIL

 Description (U) 34

 to Display (LIST) (U) 209

\$STACK

 Description (U) 31

 to Copy Lines from (B LCA Cmd) (U) .. 341

 to Copy Lines from (GET) (U) 152

 to Copy Lines from (I LCA Cmd) (U) .. 347

 to Copy Lines to (C LCA Cmd) (U) ... 342

 to Copy Lines to (K LCA Cmd) (U) ... 349

 to Copy Lines to (M LCA Cmd) (U) .. 351

 to Copy Lines to (N LCA Cmd) (U) .. 352

 to Copy Lines to (STACK) (U) 314

 to Copy Lines to (STACKI) (U) 315

 to Display (LIST) (U) 209

 to Execute (EXECUTE) (U) 128

 to Send as Mail (MAIL) (U) 229

 Tutorial (U) 8

% See Percent Sign

& See Ampersand

(See Left Parenthesis

) See Right Parenthesis

* See Asterisk

/ See Slash

; See Semicolon

? See Question Mark

@ See At Sign

\ See Backslash

~ See Tilde

| See Unbroken Vertical Bar

+ See Plus Sign

< See Less Than Sign

= See Equal Sign

> See Greater Than Sign

A LCA Command (U) 340

Access Control *See Security*

Add *See Create*

Administration

 of Auditing (U) 392

 of Checkout/Checkin (U) 394

 of Stamping (U) 388

ALT Command (ALTER) (U) 52

Alter *See Change*

ALTER Command (U) 52

ALTERL Command (U) 55

Alternate Screen

 to Set/Reset (SCREEN ALT) (U) 288

ALTERP Command

 (MVS) (U) 58

 (VSE) (U) 56

ALTL Command (ALTERL) (U) 55

ALTP Command (ALTERP) (U) 56

Ampersand (&) Command Prefix

 Overview (U) 22

 Reference (U) 333

AND (+) Pattern Character (U) 374

AP Command (ALTERP) (U) 56

Application Interface

 Overview (U) 44

Asterisk (*) Pattern Character (U) 372

At Sign (@) Pattern Character (U) 372

ATT Command (ATTACH) (U) 61

ATTACH Command (U) 61

ATTACHD Command

 (MVS) (U) 63

 (VSE) (U) 62

ATTACHX Command (U) 64

ATTD Command (ATTACHD) (U) 62

Attributes *See Member, Library, etc*

ATTX Command (ATTACHX) (U) 64

AUDCL Command (AUDITCL) (U) 65

AUDI Command (AUDITI) (U) 67

Audit	
Administration (U)	392
Discussion (U)	389
Format of Display (U)	390
to Delete (AUDITCL) (U)	65
to Display (SCREEN AUDIT) (U)	288
to Insert Line to (AUDITI) (U)	67
to Start/Stop (ALTER) (U)	52
to Summarize (AUDITSM) (U)	69
to Undo Changes (AUDITRL) (U)	68
AUDITCL Command (U)	65
AUDITI Command (U)	67
AUDITRL Command (U)	68
AUDITSM Command (U)	69
AUDRL Command (AUDITRL) (U)	68
AUDSM Command (AUDITSM) (U)	69
B Command (BOTTOM) (U)	73
B LCA Command (U)	341
BA Command (BACK) (U)	71
BACK Command (U)	71
Back out Changes (AUDITRL) (U)	68
Backslash (\) LCA Command (U)	364
Backslash (\) Pattern Character (U)	372
BACKWARD Command (BACK) (U)	71
Batch Processing	
Features Overview (U)	35
of BIM-EDIT	<i>See</i> Batch Utility
of Hexadecimal Data (U)	387
to Copy Member to (SUBMIT) (U)	317
Batch Utility (BIMUTIL)	
Overview (U)	43
BL Command (BLANK) (U)	72
BLANK Command (U)	72
BO Command (BOTTOM) (U)	73
BOT Command (BOTTOM) (U)	73
BOTTOM Command (U)	73
Bracket LCA Command	
Discussion (U)	338
to Forget (RESET) (U)	280
Break Apart Paragraph (SEPARATE) (U)	298
Break Line	
(SEPARATE) (U)	298
(SPLIT) (U)	311
C Command (CHANGE) (U)	79
C LCA Command (U)	342
Calling BIM-EDIT from Programs	<i>See</i>
Application Interface	
CANC Command (CANCEL) (U)	74
CANCEL Command (MVS) (U)	74
CAT Command (CATAL) (U)	75
CATAL Command	
(MVS) (U)	77
(VSE) (U)	75
CC LCA Command (U)	342
CEN Command (CENTER) (U)	78
CENT Command (CENTER) (U)	78
CENTER Command (U)	78
Center Lines	
(CENTER) (U)	78
(Plus Sign LCA Command) (U)	368
CH Command (CHANGE) (U)	79
Change Attributes	
of JES Job/Data Sets (ALTERP) (U)	58
of JES Job/Data Sets (FALTERP) (U) ...	135
of JES Job/Data Sets (Q LCA) (U)	354
of Library (ALTERL) (U)	55
of Library (FALTERL) (U)	132
of Library (Q LCA Command) (U)	354
of Member (ALTER) (U)	52
of Member (FALTER) (U)	130
of Member (Q LCA Command) (U)	354
of POWER Job Entry (ALTERP) (U)	56
of POWER Job Entry (FALTERP) (U) ..	133
of POWER Job Entry (Q LCA) (U)	354
of Session (FSESSION) (U)	151
of Session (SESS) (U)	302
CHANGE Command (U)	79
Change Control <i>See</i> Stamp. <i>See</i> Checkout. <i>See</i>	
Audit	
Overview (U)	40
Change Display Mode (SCREEN) (U)	288
Change Name	
of Member (FRENAME) (U)	149
of Member (MOVE) (U)	238
of Member (RENAME) (U)	274
PDS Member (RENAMED) (U)	277
Sublibrary Member (RENAMED) (U) ..	276
Change Password (PASSWORD) (U)	247
Change Position in Session	<i>See</i> Reposition
Change Text	
of Member (E LCA Command) (U)	344
of Member (EDIT) (U)	123
of PDS Member	
(E LCA Command) (U)	344
(EDITD) (U)	126
of Sublibrary Member (E LCA) (U)	344
of Sublibrary Member (EDITD) (U) ..	125
Change Text Strings (CHANGE) (U)	79

Change to Other Logical Screen		CHECKIN (U)	84
(Less Than Sign) (U)	336	CHECKOUT (U)	85
CHECKASN Command (U)	83	CHECKPUR (U)	87
CHECKIN Command		COMPARE (U)	88
Reference (U)	84	COMPILE (U)	90
CHECKOUT Command (U)	85	CONSOLEI (U)	91
Checkout/Checkin		COPY (U)	92
Administration of (U)	394	COPYD (MVS) (U)	95
Discussion (U)	393	COPYD (VSE) (U)	94
INCLUDE Processing for (U)	394	CURSOR (U)	96
CHECKPUR Command (U)	87	DA (MVS) (U)	99
CHKA Command (CHECKASN) (U)	83	DA (VSE) (U)	97
CHKI Command (CHECKIN) (U)	84	DEFINE (U)	100
CHKO Command (CHECKOUT) (U)	85	DEFINED (MVS) (U)	107
Close Session		DEFINED (VSE) (U)	105
(END) (U)	127	DEFINEL (U)	108
(SAVE) (U)	283	DELETE (U)	109
Overview (U)	28	DEMO (VSE Usage) (U)	16
CMPR Command (COMPARE) (U)	88	DISCARD (U)	110
Column		DISPLAY (U)	111
Position (%) Pattern Character (U)	375	DISPLAYC (U)	113
to Copy (PROPAGAT) (U)	253	DISPLAYD (MVS) (U)	118
to Erase (BLANK) (U)	72	DISPLAYD (VSE) (U)	116
to Erase (KEEP) (U)	183	DUP (U)	122
to Move (SHIFT) (U)	306	EDIT (U)	123
Combine Lines (FORMAT) (U)	146	EDITD (MVS) (U)	126
Command	See LCA Command	EDITD (VSE) (U)	125
ALTER (U)	52	END (U)	127
ALTERL (U)	55	Equal Sign (U)	334
ALTERP (MVS) (U)	58	EXECUTE (U)	128
ALTERP (VSE) (U)	56	FALTER (U)	130
Ampersand Prefix (U)	333	FALTERL (U)	132
ATTACH (U)	61	FALTERP (MVS) (U)	135
ATTACHD (MVS) (U)	63	FALTERP (VSE) (U)	133
ATTACHD (VSE) (U)	62	FCOPY (U)	137
ATTACHX (U)	64	FDEFINE (U)	139
AUDITCL (U)	65	FDEFINEL (U)	141
AUDITI (U)	67	FILE (SAVE) (U)	283
AUDITRL (U)	68	FIND (U)	142
AUDITSM (U)	69	FINDUP (U)	144
BACK (U)	71	FORMAT (U)	146
BLANK (U)	72	FORWARD (U)	148
BOTTOM (U)	73	FRENAME (U)	149
CANCEL (MVS) (U)	74	FSESSION (U)	151
CATAL (MVS) (U)	77	GET (U)	152
CATAL (VSE) (U)	75	GETD (MVS) (U)	154
CENTER (U)	78	GETD (VSE) (U)	153
CHANGE (U)	79	GETP (MVS) (U)	158
CHECKASN (U)	83	GETP (VSE) (U)	155

GETQ (U).....	161	NFIND (U).....	241
GROUP (U).....	162	NFINDUP (U)	243
HELP (U)	163	OPEN (U).....	245
HOLD (MVS) (U).....	167	OVERLAY (U).....	246
HOLD (VSE) (U).....	165	PASSWORD (U).....	247
INCLUDE (Ordinary) (U)	169	POSITION (U).....	248
IND\$FILE (U).....	171	PRINT (Ordinary) (U)	249
INQUIRE (U).....	173	PROCESS (U)	252
INQUIRED (VSE) (U).....	174	PROPAGAT (U).....	253
INQUIREP (MVS) (U).....	177	PURGE (U)	254
INQUIREP (VSE) (U)	175	PURGED (MVS) (U).....	257
INSERTI (U)	179	PURGED (VSE) (U)	256
JOIN (U).....	180	PURGEL (U)	258
JUSTIFYL (U)	181	PURGEP (MVS) (U).....	261
JUSTIFYR (U)	182	PURGEP (VSE) (U).....	259
KEEP (U).....	183	PURGEQ (U)	263
KEYS (U).....	184	QUALIFY (U)	264
LADD (U)	186	Question Mark (U).....	335
Less Than Sign (U)	336	QUIT (SAVE) (U).....	283
LIBDS (VSE) (U).....	192	REFRESH (U)	269
LIBRARY (U).....	187	RELEASE (MVS) (U)	272
LIBRARYD (MVS) (U)	196	RELEASE (VSE) (U).....	270
LIBRARYD (VSE) (U).....	194	RENAME (U)	274
LIBRARYL (U)	198	RENAMED (MVS) (U)	277
LIBRARYP (MVS) (U).....	202	RENAMED (VSE) (U)	276
LIBRARYP (VSE) (U)	200	RESEQ (U)	278
LIBRARYQ (U).....	204	RESET (U).....	280
LIBRARYU (U).....	205	ROLLBACK (AUDITRL) (U)	68
LIBSDL (VSE) (U)	207	ROTATE (U).....	281
LIST (U).....	209	SAVE (U).....	283
LISTD (MVS) (U)	212	SCAN (U).....	284
LISTD (VSE) (U)	210	SCREEN (U)	288
LISTP (MVS) (U).....	218	SEARCH (U).....	296
LISTP (VSE) (U)	214	SEPARATE (U)	298
LOCATE (U).....	220	SEQCHECK (U)	300
LOCATEU (U)	223	SESS (U)	302
LOGI (U)	226	SET (Ordinary) (U)	304
LOGOFF (U).....	227	SHIFT (U).....	306
LOWERCAS (U)	228	SHOW (Ordinary) (U).....	308
MAIL (U)	229	SORT (U).....	309
MAILI (U).....	231	SPLIT (U)	311
MAILSESS (U).....	233	SQUEEZE (U).....	313
MERGE (U).....	234	STACK (U).....	314
MESSAGE (U).....	236	STACKI (U)	315
MODIFY (ALTER) (U)	52	STAMPCL (U)	316
MODIFYL (ALTERL) (U)	55	SUBMIT (MVS) (U).....	319
MODIFYP (ALTERP) (U)	56	SUBMIT (VSE) (U).....	317
MOVE (U).....	238	SUBMITD (MVS) (U)	320
NEXT (U).....	240	SWAP (U)	321

TOP (U).....	322	Reference (U).....	88
UNDO (U)	323	COMPILE Command	
UP (U)	324	Reference (U).....	90
UPPERCAS (U).....	326	CONSI Command (CONSOLEI) (U)	91
VIEW (U)	327	Console Messages (DISPLAYC) (U)	113
VTOC (U)	329	CONSOLEI Command (U)	91
Command Line		Controls (Member Change) <i>See</i> Stamp. <i>See</i>	
Position on Screen (U)	15	Checkout. <i>See</i> Audit	
Commands		Overview (U)	40
Discussion		Controls (Security)	<i>See</i> Security
Customization Overview (U).....	45	Copy	
in Application Interface (U)	44	PDS Member (COPYD) (U).....	95
in Batch Utility (U)	43	Sublibrary Member (COPYD) (U)	94
in Procedures (U).....	42	Copy Column (PROPAGAT) (U)	253
Multiple		COPY Command (U)	92
Sequence of Process (U)	377	Copy Lines	
Online Help Discussion (U)	17	by Using LCA	
Rules for Entry (U)	18	Discussion (U)	338
Security (U).....	39	from \$LOG (GET) (U)	152
Sequences of (U)	379	from \$STACK (B LCA Command) (U)	341
Syntax (U)	18	from \$STACK (GET) (U).....	152
to Access Libraries (U)	25	from \$STACK (I LCA Command) (U) .	347
to Access Mail (U).....	34	from JES Job/Data Sets (GETP) (U)	158
to Access Members (U)	25	from Member (GET) (U)	152
to Access PDS Members (U).....	37	from PDS (GETD) (U)	154
to Access Sublib Members (U)	37	from POWER Job Entry (GETP) (U)	155
to Change Text (U)	33	from Sublibrary Member (GETD) (U) .	153
to Reposition Session (U).....	29	in Member (DUP) (U)	122
to Start Session (U)	27	in Member (Quote Mark LCA) (U)	363
to Stop Session (U).....	28	to \$STACK (C LCA Command) (U).....	342
List		to \$STACK (K LCA Command) (U).....	349
to Access \$STACK (U)	31	to \$STACK (M LCA Command) (U).....	351
to Access Batch Processing (U)	35	to \$STACK (N LCA Command) (U)	352
to Access Inter-User Mail (U).....	50	to \$STACK (STACK) (U)	314
to Access JES Jobs/Data Sets (U)	50	to \$STACK (STACKI) (U)	315
to Access Libraries (U)	48	Copy Member	
to Access Members (U)	47	from Member (COPY) (U)	92
to Access PDS Members (U).....	51	from Member (FCOPY) (U).....	137
to Access POWER Job Entries (U)	49	Master to Slave (CHECKOUT) (U)	85
to Access Sublib Members (U)	50	Slave to Master (CHECKIN) (U)	84
to Change Text (U)	48	Slave to Other User (CHECKASN) (U) .	83
to Reposition Session (U).....	49	to MVS PDS (CATAL) (U).....	77
Online Help Reference (HELP) (U).....	163	to VSE Sublibrary (CATAL) (U)	75
Switch Session Groups (GROUP) (U) ..	162	COPYD Command	
to Redisplay (Question Mark) (U).....	335	(MVS) (U).....	95
to Repeat (Ampersand Prefix) (U)	333	(VSE) (U).....	94
to Repeat (Equal Sign) (U).....	334	Create	
COMP Command (COMPILE) (U).....	90	Library (DEFINEL) (U)	108
COMPARE Command		Library (FDEFINEL) (U)	141

Line Numbers (RESEQ) (U)	278
Lines (A LCA Command) (U).....	340
Lines (B LCA Command) (U)	341
Lines (I LCA Command) (U)	347
Lines (LADD) (U)	186
Member (COPY) (U)	92
Member (DEFINE) (U).....	100
Member (FCOPY) (U)	137
Member (FDEFINE) (U)	139
PDS Member (DEFINED) (U)	107
Session	
Overview (U).....	27
Session EDIT-type (EDIT) (U).....	123
Session Like Current (REFRESH) (U) ..	269
Sublibrary Member (DEFINED) (U)	105
CUR Command (CURSOR) (U)	96
Currently executing POWER output	
to Display (INQUIREP) (U)	175
CURS Command (CURSOR) (U)	96
CURSOR Command (U).....	96
Cursor Control	
to Set Extended (SET) (U).....	304
Customization	
Overview (U)	45
D LCA Command (U).....	343
DA Command	
(MVS) (U).....	99
(VSE) (U).....	97
DASD VTOC (VTOC) (U)	329
Data Set.....	See JES Job or Data Sets
DC Command (DISPLAYC) (U).....	113
DD LCA Command (U)	343
DEF Command (DEFINE) (U).....	100
Default	
to Set Library (ATTACH) (U)	61
to Set PDS (ATTACHD) (U).....	63
to Set Sublibrary (ATTACHD) (U).....	62
DEFD Command (DEFINED) (U).....	105
DEFINE Command	
Reference (U).....	100
DEFINED Command	
(MVS) (U).....	107
(VSE) (U)	105
DEFINEL Command (U).....	108
DEFL Command (DEFINEL) (U)	108
DEL Command (DELETE) (U)	109
Delete	
Audit (AUDITCL) (U).....	65
Column (BLANK) (U).....	72
Column (KEEP) (U)	183
Column (SHIFT) (U).....	306
JES Job/Data Sets (DISCARD) (U)	110
JES Job/Data Sets (P LCA Cmnd) (U) .	353
JES Job/Data Sets (PURGE) (U)	261
Library (PURGEL) (U)	258
Lines (D LCA Command) (U).....	343
Lines (DELETE) (U).....	109
Mail (DISCARD) (U)	110
Mail (P LCA Command) (U).....	353
Mail (PURGEQ) (U).....	263
Member (P LCA Command) (U)	353
Member (PURGE) (U)	254
PDS Member (P LCA Command) (U)..	353
PDS Member (PURGED) (U)	257
POWER Job Entry (DISCARD) (U)	110
POWER Job Entry (P LCA Cmnd) (U).353	
POWER Job Entry (PURGE) (U)	259
Slave Member (CHECKPUR) (U).....	87
Sublibrary Member (P LCA Cmnd) (U)353	
Sublibrary Member (PURGED) (U)	256
DELETE Command (U).....	109
DEMO Procedure (VSE)	
Usage (U)	16
DI Command (DISPLAY) (U)	111
DID Command (DISPLAYD) (U).....	116
DISC Command (DISCARD) (U)	110
DISCARD Command (U)	110
DISP Command (DISPLAY) (U).....	111
DISPC Command (DISPLAYC) (U).....	113
DISPD Command (DISPLAYD) (U)	116
Display (Screen)	
Layout (U).....	15
to Change Mode (SCREEN) (U).....	288
Display Attributes	
of JES Job (INQUIREP) (U).....	177
of JES Job (X LCA Command) (U).....	360
of Member (INQUIRE) (U).....	173
DISPLAY Command (U)	111
Display Information About Commands	
(HELP) (U).....	163
Display Line on Consol(CONSOLEI) (U) ..	91
Display Lines Containing Pattn	
(QUALIFY) (U)	264
Display List	
of JES Jobs/Data Set(LIBRARYP) (U) ..	202
of Libraries (LIBRARYL) (U).....	198
of Mail (LIBRARYQ) (U).....	204
of Members (LIBRARY) (U)	187

of PDS Members (LIBRARYD) (U)	196	EDIT Command (U)	123
of POWER Job Entries(LIBRARYP) (U)	200	EDITD Command (MVS) (U)	126
of Sublibrary Members (LIBDS (U)	192	(VSE) (U)	125
(LIBRARYD) (U)	194	Editing	386
of Users (LIBRARYU) (U)	205	Hexadecimal (U)	386
Display Status of System (DA) (U)	97	Members Overview (U)	32
Display Text of Audit Trail (SCREEN AUDIT) (U) ..	288	Tutorial (U)	5
of ICCF Member (L LCA Command) (U)	350	PDS Members Overview (U)	32
(T LCA Command) (U)	357	Sublibrary Members Overview (U)	32
of JES Job/Data Sets (L LCA) (U)	350	EDIT-type Sessions Overview (U)	27
of JES Jobs/Data Sets (LISTP) (U)	218	EE LCA Command (U)	344
of Mail (L LCA Command) (U)	350	End	<i>See Stop</i>
of Mail (OPEN) (U)	245	END Command (U)	127
of Mbrs Containing Pttrn(SCAN) (U) ..	284	Entering BIM-EDIT (U)	14
of Member (DISPLAY) (U)	111	Entering Commands Rules for (U)	18
of Member (L LCA Command) (U)	350	Equal Sign Command Overview (U)	22
of Member (LIST) (U)	209	Reference (U)	334
of Member (T LCA Command) (U)	357	Equal Sign in a Command Operand (U)	19
of PDS Member (DISPLAYD) (U)	118	EX Command (EXECUTE) (U)	128
(L LCA Command) (U)	350	Exchange Position Markers (SWAP) (U) ..	321
(LISTD) (U)	212	EXEC Command (EXECUTE) (U)	128
(T LCA Command) (U)	357	EXECUTE Command (U)	128
of POWER Job Entry (L LCA) (U)	350	Exiting BIM-EDIT (U)	14
of POWER Job Entry (LISTP) (U)	214	Explain Support (MESSAGE) (U)	236
of Sublibrary Mem (DISPLAYD) (U) ...	116	Extended Search Pattern Character Definitions (U)	372
of Sublibrary Member (L LCA) (U)	350	Combination Searches (U)	374
of Sublibrary Member (LISTD) (U)	210	Finding First Non-Blank (U)	376
of Sublibrary Member (T LCA) (U)	357	Overview (U)	372
Display VSE Console Messages (DISPLAYC) (U)	113	Setting Position (U)	375
DISPLAYC Command (U)	113	to Change (CHANGE) (U)	79
DISPLAYD Command (MVS) (U)	118	to Display List (QUALIFY) (U)	264
(VSE) (U)	116	to Display Mbrs having (SCAN) (U) ...	284
DISP-type Sessions Overview (U)	27	to Search for (FIND) (U)	142
DO Command (NEXT) (U)	240	to Search for (FINDUP) (U)	144
DOWN Command (NEXT) (U)	240	to Search for (LOCATE) (U)	220
DQ Command (LIBRARYP) (U)	200	to Search for (LOCATEU) (U)	223
DUP Command (U)	122	to Search for (NFIND) (U)	241
Duplicate	<i>See Copy</i>	to Search for (NFINDUP) (U)	243
E LCA Command (U)	344	to Search for (SEARCH) (U)	296
ED Command (EDIT) (U)	123	to Search for Words (U)	373
EDD Command (EDITD) (U)	125	to Use with CHANGE (U)	375
		to Use with CURSOR (U)	375

Wildcard Searches (U)	374	Hexadecimal	
F Command (FIND) (U)	142	Display and Edit Discussion (U)	385
FALT Command (FALTER) (U)	130	Substitution during SUBMIT (U)	387
FALTER Command (U)	130	to Display (SCREEN HEX) (U)	288
FALTERL Command (U)	132	to Set Escape Char (SET) (U)	304
FALTERP Command		HH LCA Command (U)	346
(MVS) (U)	135	HOLD Command	
(VSE) (U)	133	(MVS) (U)	167
FALTL Command (FALTERL) (U)	132	(VSE) (U)	165
FALTP Command (FALTERP) (U)	133	Home	
FCOPY Command (U)	137	to Set Library (ATTACH) (U)	61
FDEF Command (FDEFINE) (U)	139	to Set PDS (ATTACHD) (U)	63
FDEFINE Command (U)	139	to Set Sublibrary (ATTACHD) (U)	62
FDEFINEL Command (U)	141	I LCA Command (U)	347
FDEFL Command (FDEFINEL) (U)	141	ICCF	
FILE Command (SAVE) (U)	283	to Display Text (L LCA Cmnd) (U)	350
FIND Command (U)	142	to Display Text (T LCA Cmnd) (U)	357
FINDU Command (FINDUP) (U)	144	IESMSGs Support (MESSAGE) (U)	236
FINDUP Command (U)	144	INCLUDE Command	
FO Command (FORWARD) (U)	148	(Ordinary) (U)	169
FOR Command (FORWARD) (U)	148	with Checked Out Members (U)	394
FORM Command (FORMAT) (U)	146	IND\$FILE Command (U)	171
FORMAT Command (U)	146	Initiate	<i>See Start</i>
FORWARD Command (U)	148	INQ Command (INQUIRE) (U)	173
FREN Command (FRENAME) (U)	149	INQD Command (INQUIRED) (U)	174
FRENAME Command (U)	149	INQP Command (INQUIREP) (U)	175, 177
FSESSION Command (U)	151	Inquire	
FU Command (FINDUP) (U)	144	Sublibrary Member (INQUIRED) (U) ..	174
FUP Command (FINDUP) (U)	144	INQUIRE Command (U)	173
G LCA Command (U)	345	INQUIRED Command (VSE) (U)	174
GET Command (U)	152	INQUIREP Command (MVS) (U)	177
GETD Command		INQUIREP Command (VSE) (U)	175
(MVS) (U)	154	INS Command (INSERTI) (U)	179
(VSE) (U)	153	INSERT Command (INSERTI) (U)	179
GETP Command		Insert Line	
(MVS) (U)	158	from \$STACK (B LCA Command) (U) ..	341
(VSE) (U)	155	from \$STACK (I LCA Command) (U) ..	347
GETQ Command (U)	161	from JES Job/Data Sets (GETP) (U) ..	158
Greater Than Sign (>)		from Mail (GETQ) (U)	161
LCA Command (U)	367	from PDS Member (GETD) (U)	154
Pattern Character (U)	372	from POWER Job Entry (GETP) (U) ..	155
Group		from Sublibrary Member (GETD) (U) ..	153
Overview (U)	28	to \$LOG (LOGI) (U)	226
GROUP Command		to \$STACK (STACKI) (U)	315
Reference (U)	162	to Audit (AUDITI) (U)	67
H LCA Command (U)	346	to Console (CONSOLEI) (U)	91
HELP Command		to Session (A LCA Command) (U) ..	340
Overview (U)	17	to Session (GET) (U)	152
Reference (U)	163	to Session (INSERTI) (U)	179

INSERTI Command (U)	179	KK LCA Command (U)	349
INSI Command (INSERTI) (U)	179	L Command (LOCATE) (U)	220
Interface to BIM-EDIT <i>See</i> Application Interface		L LCA Command (U)	350
Inter-User Mail	<i>See</i> Mail	LA Command (LADD) (U)	186
J LCA Command (U)	348	LADD Command (U)	186
JES		Last Referenced	
to Copy Member to (SUBMIT) (U)	319	Entities in Command (U)	21
to Copy PDS Mbr to (SUBMITD) (U) ..	320	LCA	
JES Job or Data Sets		to Change Position (SCREEN LCA) (U) ..	288
Access Overview (U)	35	LCA Command	
Last Referenced		A (U)	340
in Command (U)	21	B (U)	341
List of Commands to Access (U)	50	Backslash (\) (U)	364
Security Overview (U)	39	C,CC (U)	342
to Change Attr		D,DD (U)	343
(ALTERP) (U)	58	E,EE (U)	344
(FALTERP) (U)	135	G (U)	345
(Q LCA Command) (U)	354	Greater Than Sign (>) (U)	367
to Delete (DISCARD) (U)	110	H,HH (U)	346
to Delete (P LCA Command) (U)	353	Help Online (U)	17
to Delete (PURGE) (U)	261	I (U)	347
to Display Attr (INQUIREP) (U)	177	J,JJ (U)	348
to Display Attr (X LCA Command) (U) ..	360	K,KK (U)	349
to Display List (LIBRARYP) (U)	202	L,LL (U)	350
to Display Text (L LCA Cmnd) (U)	350	Left Parenthesis (U)	369
to Display Text (LISTP) (U)	218	Less Than Sign (<) (U)	366
to Insert Lines from (GETP) (U)	158	M,MM (U)	351
to Stop (CANCEL) (U)	74	N,NN (U)	352
to Suspend (H LCA Command) (U)	346	P,PP (U)	353
to Suspend (HOLD) (U)	167	Plus Sign (+) (U)	368
to Undo Hold (R LCA Command) (U) ..	355	Q,QQ Command (U)	354
to Undo Hold (RELEASE) (U)	272	Quote Mark (U)	363
JJ LCA Command (U)	348	R,RR (U)	355
Job Entry	<i>See</i> POWER Job Entry	Right Parenthesis (U)	370
Job Notification		S,SS (U)	356
Introduction (U)	36	Slash (/) (U)	365
SUBMIT (NTFY) (U)	317	T,TT (U)	357
JOIN Command (U)	180	U,UU (U)	358
Join two lines of text (U)	180	W,WW (U)	359
JUSTIFYL Command (U)	181	X,XX (U)	360
JUSTIFYR Command (U)	182	Y (U)	362
JUSTL Command (JUSTIFYL) (U)	181	LCA Commands	
JSTR Command (JUSTIFYR) (U)	182	Bracket (U)	338
K LCA Command (U)	349	Discussion (U)	337
KEEP Command (U)	183	Double Letter (U)	338
KEY Command (KEYS) (U)	184	for Accessing \$STACK (U)	31
KEYS Command (U)	184	for Editing	
Keyword Operands		Overview (U)	32
Rules for Entry (U)	19	Multiple on a Screen (U)	339

on LIBRARYx Sessions (U)	30	LIBRARYD Command	
Position on Screen (U).....	15	(MVS) (U).....	196
Repeat Count (U).....	338	(VSE) (U).....	194
Sequence of Processing (U)	339	LIBRARYL Command (U)	198
to Forget Bracket (RESET) (U)	280	LIBRARYP Command	
to Set Default Position (SET) (U)	304	(MVS) (U).....	202
Tutorial (U).....	5	(VSE) (U).....	200
Left Justify Line		LIBRARYQ Command (U).....	204
(JUSTIFYL) (U).....	181	LIBRARYU Command (U).....	205
(Left Parenthesis LCA Command) (U)	369	LIBRARYx Sessions	
Left Parenthesis		Use of LCA (U).....	30
LCA Command (U).....	369	LIBSDL Command	
Left Shift Column		(VSE) (U).....	207
(Less Than Sign LCA Command) (U)..	366	LIBU Command (LIBRARYU) (U).....	205
(SHIFT) (U).....	306	LID Command (LISTD) (U)	210
Left Shift Display (VIEW) (U).....	327	Line	
Less Than Sign (<)		to Break Apart (SEPARATE) (U)	298
Command Overview (U).....	22	to Break Apart (SPLIT) (U)	311
Command Reference (U)	336	to Center (CENTER) (U)	78
LCA Command (U).....	366	to Center (Plus Sign LCA Cmnd) (U) ..	368
Pattern Character (U)	372	to Combine (FORMAT) (U).....	146
LI Command (LIST) (U)	209	to Copy (C LCA Command) (U).....	342
LIB Command (LIBRARY) (U).....	187	to Copy (K LCA Command) (U)	349
LIBD Command (LIBRARYD) (U).....	194	to Create (A LCA Command) (U)	340
LIBDS Command		to Create (LADD) (U).....	186
(VSE) (U).....	192	to Delete (D LCA Command) (U).....	343
LIBL Command (LIBRARYL) (U)	198	to Delete (DELETE) (U).....	109
LIBP Command (LIBRARYP) (U)	200	to Duplicate (DUP) (U)	122
LIBQ Command (LIBRARYQ) (U).....	204	to Duplicate (Quote Mark LCA) (U) ...	363
Library		to Insert (INSERTI) (U)	179
Discussion (U).....	25	to Insert from \$STACK (B LCA) (U) ...	341
List of Commands which Access (U).....	48	to Insert from \$STACK (I LCA) (U)	347
Rules for Referencing (U)	26	to Insert to \$LOG (LOGI) (U)	226
to Change Attributes		to Insert to \$STACK (STACKI) (U).....	315
(ALTERL) (U)	55	to Insert to Audit (AUDITI) (U).....	67
(FALTERL) (U).....	132	to Insert to Console(CONSOLEI) (U).....	91
(Q LCA) (U)	354	to Left Just (JUSTIFYL) (U).....	181
to Create (DEFINEL) (U)	108	to Left Just (Left Paren LCA) (U).....	369
to Create (FDEFINEL) (U)	141	to Make Lower Case (LOWERCAS) (U)228	
to Delete (PURGEL) (U).....	258	to Make Lower Case (W LCA) (U)	359
to Display List (LIBRARYL) (U)	198	to Make Upper Case (U LCA) (U)	358
to Display Mbrs w/ Ptrn (SCAN) (U) .	284	to Make Upper Case (UPPERCAS) (U)326	
to Move Member (FRENAME) (U)	149	to Merge (G LCA Command) (U).....	345
to Move Member (MOVE) (U).....	238	to Merge (J LCA Command) (U)	348
to Move Member (RENAME) (U)	274	to Merge (MERGE) (U)	234
to Set Default (ATTACH) (U)	61	to Merge (OVERLAY) (U)	246
to Set Default (Y LCA Command) (U) .	362	to Move (M LCA Command) (U)	351
to Set Home (ATTACH) (U)	61	to Move (N LCA Command) (U).....	352
LIBRARY Command (U).....	187	to Number (RESEQ) (U)	278

to Right Just (JUSTIFYR) (U).....	182	to Delete (DISCARD) (U)	110
to Right Just(Right Paren LCA) (U)	370	to Delete (P LCA Command) (U)	353
to Shift (SHIFT) (U)	306	to Delete (PURGEQ) (U).....	263
to Shift Left (Less Than Sign) (U)	366	to Display List (LIBRARYQ) (U)	204
to Shift Right(Grtr Than Sign) (U).....	367	to Insert Lines from (GETQ) (U).....	161
to Sort (SORT) (U)	309	to Receive (L LCA Command) (U).....	350
Line Control Area	See LCA	to Receive (OPEN) (U)	245
List.....	See Display	to Send (MAIL) (U)	229
LIST Command (U).....	209	to Send (MAILI) (U)	231
LISTD Command		to Send Session (MAILSESS) (U).....	233
(MVS) (U)	212	to Set Proxy User for(ATTACHX) (U) ...	64
(VSE) (U).....	210	Mail Area.....	See \$MAIL
LISTD-type Sessions Overview (U)	27	MAIL Command (U)	229
LISTP Command		MAILI Command (U)	231
(MVS) (U)	218	MAILSESS Command (U).....	233
(VSE) (U).....	214	MAIL-type Sessions Overview (U)	28
LISTP-type Sessions Overview (U).....	27	Make.....	See Create
LIST-type Sessions Overview (U)	27	Master Member in Checkout	
LL LCA Command (U).....	350	to Copy from Slave (CHECKIN) (U)	84
LOC Command (LOCATE) (U)	220	to Copy to Slave (CHECKOUT) (U).....	85
LOCATE Command (U)	220	Member	
LOCATEU Command (U)	223	Auditing Discussion (U).....	389
LOCU Command (LOCATEU) (U)	223	Change Controls	
LOCUP Command (LOCATEU) (U).....	223	Overview (U).....	40
Log Area.....	See \$LOG	Checkout/Checkin Discussion (U)	393
Logging On and Logging Off (U)	14	Creating	
LOGI Command (U)	226	Tutorial (U)	4
Logical Screens		Discussion (U).....	25
Discussion (U).....	381	Editing	
Logical Tabbing		Tutorial (U)	5
Discussion (U).....	383	Editing Overview (U)	32
to Set Character (SET) (U)	304	Last Referenced	
LOGOF Command (LOGOFF) (U)	227	in Command (U)	21
LOGOFF Command (U).....	227	List of Attributes (U)	25
LOW Command (LOWERCAS) (U)	228	List of Commands to Access (U)	47
Lower Case Translate		Rules for Referencing (U)	26
Line (LOWERCAS) (U).....	228	Stamping Discussion (U).....	388
Line (W LCA Command) (U)	359	to Change Attr	
LOWER Command (LOWERCAS) (U)	228	(ALTER) (U)	52
LOWERCAS Command (U)	228	(FALTER) (U)	130
LP Command (LISTP) (U).....	214	(Q LCA Command) (U)	354
LU Command (LOCATEU) (U)	223	to Change Name (FRENAME) (U).....	149
LUP Command (LOCATEU) (U)	223	to Change Name (MOVE) (U)	238
M LCA Command (U).....	351	to Change Name (RENAME) (U)	274
Macros	See Procedures	to Change Position	See Reposition
Mail		to Change Text (E LCA Command) (U).....	344
List of Commands to Access (U)	50	to Change Text (EDIT) (U)	123
Overview (U)	34	to Compile (COMPILE) (U)	90
Security Overview (U).....	39	to Copy (COPY) (U)	92

to Copy (FCOPY) (U)	137
to Copy Mastr->Slave(CHECKOUT) (U).....	85
to Copy Slave->Master(CHECKIN) (U)	84
to Copy Slave->User (CHECKASN) (U)	83
to Copy to PDS (CATAL) (U)	77
to Copy to Sublibrary (CATAL) (U)	75
to Create (DEFINE) (U).....	100
to Create (FDEFINE) (U)	139
to Delete (P LCA Command) (U)	353
to Delete (PURGE) (U).....	254
to Display Attr (INQUIRE) (U).....	173
to Display List (LIBRARY) (U)	187
to Display Text (DISPLAY) (U)	111
to Display Text (L LCA Cmnd) (U).....	350
to Display Text (LIST) (U)	209
to Display Text (T LCA Cmnd) (U).....	357
to Find with Pattern (SCAN) (U).....	284
to Include in SUBMIT (INCLUDE) (U)	169
to Start Batch Proces(PROCESS) (U)....	252
to Start Batch Process (S LCA) (U)	356
to Start Batch Process(SUBMIT) (U).....	317
to Undo Changes (AUDITRL) (U)	68
Member Security (U)	38
Members	
to Compare (COMPARE) (U)	88
MERGE Command (U).....	234
Merge Line	
(G LCA Command) (U)	345
(J LCA Command) (U).....	348
(MERGE) (U)	234
(OVERLAY) (U)	246
MESSAGE Command (U)	236
Message Line	
Position on Screen (U).....	15
MM LCA Command (U)	351
MOD Command (ALTER) (U)	52
Modify	<i>See Change</i>
MODIFY Command (ALTER) (U)	52
MODIFYL Command (ALTERL) (U)	55
MODIFYP Command (ALTERP) (U)	56
MODL Command (ALTERL) (U)	55
MODP Command (ALTERP) (U).....	56
Move	<i>See Copy</i>
Move Column	
(PROPAGAT) (U)	253
(SHIFT) (U).....	306
MOVE Command (U).....	238
Move Lines	
(M LCA Command) (U)	351

(N LCA Command) (U)	352
using LCA	
Discussion (U)	338
Move Member	
Slave to Other User (CHECKASN) (U) .	83
to Another Library (FRENAME) (U) ...	149
to Another Library (MOVE) (U)	238
to Another Library (RENAME) (U).....	274
Move Position in Session.....	<i>See Reposition</i>
MSG Command(MESSAGE) (U).....	236
Multiple	
Commands	
Rules for Entry (U).....	22
Techniques for Use (U)	379
LCA Commands on a Screen (U)	339
Members on a Screen (U).....	381
Operations on a Screen (U).....	377
Sessions	
Overview (U).....	28
Windows on a Screen (U)	381
MVS.....	<i>See also JES</i>
MVS PDS	<i>See PDS</i>
N Command (NEXT) (U)	240
N LCA Command (U)	352
Name=Value Operands	
Rules for Entry (U)	19
Names	
of Commands	
Rules for Entry (U).....	18
of Members and Libraries (U).....	26
of Operands (U)	19
NEXT Command (U)	240
NF Command (NFIND) (U).....	241
NFIND Command (U).....	241
NFINDU Command (NFINDU) (U)	243
NFINDUP Command (U)	243
NFU Command (NFINDUP) (U)	243
NFUP Command (NFINDUP) (U).....	243
NN LCA Command (U)	352
NOT (~) Pattern Character (U)	374
Number Lines (RESEQ) (U)	278
O Command (OVERLAY) (U)	246
Online Help Discussion (U)	17
OP Command (OPEN) (U).....	245
OPEN Command (U)	245
Open Session	
Overview (U).....	27
Operands	
On/Off values (U)	20

Operands - Rules for Entry (U)	18
OR () Pattern Character (U)	374
Order Lines (SEQCHECK) (U)	300
Order Lines (SORT) (U)	309
OVERLAY Command (U)	246
Overview of BIM-EDIT Functions (U)	1
OVLY Command (OVERLAY) (U)	246
P Command (POSITION) (U)	248
P LCA Command (U)	353
Pan Display	<i>See Reposition</i>
Paragraph	
to Break Apart (SEPARATE) (U)	298
to Put Together (FORMAT) (U)	146
Partitioned Data Set	<i>See PDS</i>
PAS Command (PASSWORD) (U)	247
PASS Command (PASSWORD) (U)	247
PASSWORD Command (U)	247
Patterns	<i>See Extended Search Pattern</i>
PC File Transfer (IND\$FILE) (U)	171
PDS (MVS)	
to Set Default (ATTACHD) (U)	63
to Set HOME (ATTACHD) (U)	63
PDS (MVS) Member	
Access Overview (U)	37
Editing Overview (U)	32
Last Referenced	
in Command (U)	21
List of Commands to Access (U)	51
Security Discussion (U)	39
to Change Name (RENAMED) (U)	277
to Change Text (E LCA Command) (U)	344
to Change Text (EDITD) (U)	126
to Copy (COPYD) (U)	95
to Copy from Member (CATAL) (U)	77
to Create (DEFINED) (U)	107
to Delete (P LCA Command) (U)	353
to Delete (PURGED) (U)	257
to Display List (LIBRARYD) (U)	196
to Display Text (DISPLAYD) (U)	118
to Display Text (L LCA Cmd) (U)	350
to Display Text (LISTD) (U)	212
to Display Text (T LCA Cmd) (U)	357
to Get Lines from (GETD) (U)	154
to Start Batch Proces (SUBMITD) (U)	320
Percent Sign (%) Pptrn Character (U)	372
PF and PA Keys	
for Commands	
Overview (U)	22
to Set (KEYS) (U)	184
to Set (SET) (U)	304
Plus Sign (+)	
LCA Command (U)	368
Pattern Character (U)	372
POS Command (POSITION) (U)	248
Position (%) Pattern Character (U)	375
POSITION Command (U)	248
Position Cursor (CURSOR) (U)	96
Position in a Session	<i>See Reposition</i>
Positional Operands	
Rules for Entry (U)	18
POWER	<i>See also VSE</i>
to Copy Member to (SUBMIT) (U)	317
POWER Job Entry	
Access Overview (U)	35
Last Referenced	
in Command (U)	21
List of Commands to Access (U)	49
Security Overview (U)	39
to Change Attr	
(ALTERP) (U)	56
(FALTERP) (U)	133
(Q LCA Command) (U)	354
to Delete (DISCARD) (U)	110
to Delete (P LCA Command) (U)	353
to Delete (PURGE) (U)	259
to Display List (LIBRARYP) (U)	200
to Display Text (L LCA Cmd) (U)	350
to Display Text (LISTP) (U)	214
to Insert Lines from (GETP) (U)	155
to Suspend (H LCA Command) (U)	346
to Suspend (HOLD) (U)	165
to Undo Hold (R LCA Command) (U)	355
to Undo Hold (RELEASE) (U)	270
PP Command (PURGE) (U)	259
PP LCA Command (U)	353
PRINT Command	
(Ordinary) (U)	249
PROC Command (PROCESS) (U)	252
Procedures	
Overview (U)	41
Rules for Entering Commands (U)	23
to Execute (EXECUTE) (U)	128
PROCESS Command	
Reference (U)	252
Processing	
Batch Features Discussion (U)	35
Programs	

Calling BIM-EDIT from See Application Interface	
Display Status of(DA) (U)	97
PROP Command (PROPAGAT) (U)	253
PROPAGAT Command (U)	253
PSWD	See Member Security
PSWD Command (PASSWORD) (U)	247
PUR Command (PURGE) (U)	254
PURD Command (PURGED) (U)	256
Purge	See Delete
PURGE Command (U)	254
PURGED Command	
(MVS) (U)	257
(VSE) (U)	256
PURGEL Command (U)	258
PURGE Command	
(MVS) (U)	261
(VSE) (U)	259
PURGEQ Command (U)	263
PURL Command (PURGEL) (U)	258
PURP Command (PURGE) (U)	259
PURQ Command (PURGEQ) (U)	263
Q Command (SAVE) (U)	283
Q LCA Command (U)	354
QQ LCA Command (U)	354
QUAL Command (QUALIFY) (U)	264
QUALIFY Command (U)	264
Question Mark (?)	
Command Overview (U)	22
Command Reference (U)	335
Pattern Character (U)	372
QUIT Command (SAVE) (U)	283
Quote Mark	
LCA Command (U)	363
R LCA Command (U)	355
Receive Mail (OPEN) (U)	245
Redisplay	
Command (Question Mark) (U)	335
Session (REFRESH) (U)	269
REF Command (REFRESH) (U)	269
REFRESH Command	
Reference (U)	269
REL Command (RELEASE) (U)	270
RELEASE Command	
(MVS) (U)	272
(VSE) (U)	270
RELIST Command (PRINT) (U)	249
Remove	See Delete
Remove embedded blanks	
(SQUEEZE) (U)	313
REN Command (RENAME) (U)	274
RENAME Command (U)	274
RENAMED Command	
(MVS) (U)	277
(VSE) (U)	276
REND Command (RENAMED) (U)	276
RENUM Command (RESEQ) (U)	278
Reorder Lines (SORT) (U)	309
Repeat Command	
(Ampersand Command Prefix) (U)	333
(Equal Sign) (U)	334
Repeat Count on LCA Commands (U)	338
Replace	See Change
Reposition	
Bwd by Lines (UP) (U)	324
Bwd by Screens (BACK) (U)	71
Bwd to Patrn (LOCATEU) (U)	223
Bwd to Patrn at Col (FINDUP) (U)	144
Bwd to Patrn at Col (NFINDUP) (U)	243
Fwd by Lines (NEXT) (U)	240
Fwd by Screens (FORWARD) (U)	148
Fwd to Patrn (LOCATE) (U)	220
Fwd to Patrn (SEARCH) (U)	296
Fwd to Patrn at Col (FIND) (U)	142
Fwd to Patrn at Col (NFIND) (U)	241
Line (Backslash LCA Command) (U)	364
Overview (U)	29
to Bottom (BOTTOM) (U)	73
to Left or Right (VIEW) (U)	327
to Line (POSITION) (U)	248
to Line (Slash LCA Command) (U)	365
to Marker (SWAP) (U)	321
to Top (TOP) (U)	322
RES Command (RESEQ) (U)	278
RESE Command (RESEQ) (U)	278
RESEQ Command (U)	278
RESET Command (U)	280
Resume JES Job or Data Sets	
(R LCA Command) (U)	355
(RELEASE) (U)	272
Resume POWER Job Entry	
(R LCA Command) (U)	355
(RELEASE) (U)	270
Revise	See Change
Right Justify Line	
(JUSTIFYR) (U)	182
(Right Paren LCA Command) (U)	370
Right Parenthesis LCA Command (U)	370

Right Shift Column	
(Greater Than LCA Command) (U).....	367
(SHIFT) (U).....	306
Right Shift Display (VIEW) (U).....	327
ROLLBACK Command (AUDITRL) (U) ...	68
ROT Command (ROTATE) (U).....	281
ROTATE Command (U).....	281
RR LCA Command (U).....	355
S LCA Command (U).....	356
SAVE Command (U).....	283
Scale Line	
Position on Screen (U).....	15
to Change Position(SCREEN BEF) (U)	288
SCAN Command (U).....	284
SCR Command (SCREEN) (U).....	288
SCREEN Command (U).....	288
Screen Layout (U).....	15
Screen Split Discussion (U).....	381
Scroll.....	<i>See Reposition</i>
SDL Display LIBSDL (U).....	207
SEARCH Command (U).....	296
Search for Text.. <i>See Extended Search Pattern</i>	
(LOCATE) (U).....	220
(LOCATEU) (U).....	223
(SEARCH) (U).....	296
at Column (FIND) (U).....	142
at Column (FINDUP) (U).....	144
list Instances (QUALIFY) (U).....	264
not at Column (NFIND) (U).....	241
not at Column (NFINDUP) (U).....	243
Security	
for Libraries	
Access Levels (U).....	38
for Members (U).....	38
Overview (U).....	38
See Index.....	<i>See Thing</i>
Select Session (ROTATE) (U).....	281
Semicolon	
to Separate Command & Comment (U)	23
to Separate Online Commands (U).....	22
Send Mail	
(MAIL) (U).....	229
(MAILI) (U).....	231
Current Session (MAILSESS) (U).....	233
SEP Command (SEPARATE) (U).....	298
SEPARATE Command (U).....	298
SEQCHECK Command (U).....	300
Sequence Check Lines (SEQCHECK) (U)	300
Sequence Lines (SORT) (U).....	309
Sequence of Processing	
LCA Commands (U).....	339
Multiple Operations (U).....	377
Sequences of Commands	
Techniques for Use (U).....	379
SESS Command (U).....	302
Session	
Commands to Change Text (U).....	48
Commands to Reposition (U).....	49
Editing Usage Overview (U).....	32
Multiple	
Overview (U).....	28
Overview (U).....	27
to Change Attributes	
(FSESSION) (U).....	151
(SESS) (U).....	302
to Change Position.....	<i>See Reposition</i>
to Execute (EXECUTE) (U).....	128
to Re-create (REFRESH) (U).....	269
to Send as Mail (MAIL) (U).....	229
to Send as Mail (MAILSESS) (U).....	233
to Stop (END) (U).....	127
to Switch (ROTATE) (U).....	281
to Terminate (SAVE) (U).....	283
Sessions	
Switch Groups (GROUP) (U).....	162
SET Command	
(Ordinary) (U).....	304
SH Command (SHIFT) (U).....	306
SHIFT Command (U).....	306
Shift Display (VIEW) (U).....	327
Shift Lines Left	
(Less Than Sign LCA Command) (U)..	366
Shift Lines Right	
(Greater Than Sign LCA Cmnd) (U)....	367
SHOW Command	
(Ordinary) (U).....	308
Slash (/) LCA Command (U).....	365
Slave Member in Checkout	
to Change User (CHECKASN) (U).....	83
to Copy from Master(CHECKOUT) (U)	85
to Copy to Master (CHECKIN) (U).....	84
to Delete (CHECKPUR) (U).....	87
SORT Command (U).....	309
SP Command (SPLIT) (U).....	311
SPLIT Command (U).....	311
Split Screen	
Discussion (U).....	381
to Chg Screens(Less Than Sign) (U)....	336

to Start/Stop (SCREEN SPLIT) (U).....	288
SQU Command (SQUEEZE) (U).....	313
SQUEEZE Command (U).....	313
SQUEZ Command (SQUEEZE) (U).....	313
SRCH Command (SEARCH) (U)	296
SS Command (SESS) (U)	302
SS LCA Command (U)	356
ST Command (STACK) (U)	314
Stack Area	see \$STACK
STACK Command (U).....	314
STACKI Command (U)	315
STAMPCL Command (U).....	316
Stamping	
Administration of (U)	388
Discussion (U).....	388
to Clear (STAMPCL) (U)	316
to Display (SCREEN STAMP) (U).....	288
to Start/Stop (ALTER) (U)	52
Start Audit (ALTER) (U)	52
Start Batch Processing	
(PROCESS) (U).....	252
(S LCA Command) (U)	356
(SUBMIT) (U)	317
(SUBMITD) (U)	320
Start Session	
Overview (U)	27
Start Stamping (ALTER) (U).....	52
Status	
of Running Programs (DA) (U)	97
Stop Audit (ALTER) (U).....	52
Stop JES Job (CANCEL) (U).....	74
Stop Stamping (ALTER) (U)	52
Stop Using BIM-EDIT	
(LOGOFF) (U)	227
Stop Using Session	
(END) (U)	127
(SAVE) (U).....	283
Stopping a Session	
Overview (U)	28
Tutorial (U).....	12
String.....	See Text String
SUB Command (SUBMIT) (U).....	317
SUBD Command (SUBMITD) (U)	320
Sublibrary (VSE)	
to Set Default (ATTACHD) (U)	62
to Set Home (ATTACHD) (U)	62
Sublibrary (VSE) Member	
Access Overview (U).....	37
Editing Overview (U)	32

Last Referenced	
in Command (U)	21
List of Commands to Access (U)	50
Security Discussion (U).....	39
to Change Name (RENAMED) (U)	276
to Change Text (E LCA Command) (U).....	344
to Change Text (EDITD) (U)	125
to Copy (COPYD) (U)	94
to Copy fm Member (CATAL) (U).....	75
to Create (DEFINED) (U).....	105
to Delete (P LCA Cmnd) (U).....	353
to Delete (PURGED) (U)	256
to Display List (LIBDS) (U).....	192
to Display List (LIBRARYD) (U)	194
to Display Text (DISPLAYD) (U).....	116
to Display Text (L LCA Cmnd) (U).....	350
to Display Text (LISTD) (U)	210
to Display Text (T LCA Cmnd) (U).....	357
to Inquire (INQUIRED) (U).....	174
to Insert Lines from (GETD) (U).....	153
SUBMIT Command	
(MVS) (U).....	319
(VSE) (U).....	317
SUBMITD Command (MVS) (U).....	320
Summarize Audit Trail (AUDITSM) (U)....	69
Summary of BIM-EDIT Functions (U).....	1
Suspend JES Job/Data Sets	
(H LCA Command) (U)	346
(HOLD) (U)	167
Suspend POWER Job Entry	
(H LCA Command) (U)	346
(HOLD) (U)	165
SW Command (SWAP) (U).....	321
SWAP Command (U).....	321
Switch Position Markers (SWAP) (U).....	321
Switch Session Groups (GROUP) (U).....	162
Switch Sessions (ROTATE) (U)	281
Syntax	
Commands (U).....	18
Member and Library Names (U)	26
T Command (TOP) (U)	322
T LCA Command (U)	357
Tabbing	
Discussion (U)	383
Tall Display	See Alternate Screen
Terminate	See Stop
Text Display Area	
Position on Screen (U).....	15
Text String	

to Change (CHANGE) (U)	79
to Display List (QUALIFY) (U)	264
to Display Mbrs having (SCAN) (U) ...	284
to Search for (FIND) (U)	142
to Search for (FINDUP) (U)	144
to Search for (LOCATE) (U)	220
to Search for (LOCATEU) (U)	223
to Search for (NFIND) (U)	241
to Search for (NFINDUP) (U)	243
to Search for (SEARCH) (U)	296
Tilde (~) Pattern Character (U)	372
TOP Command (U)	322
Transfer	<i>See Copy</i>
Translate to Lower Case	
Line (LOWERCAS) (U)	228
Line (W LCA Command) (U)	359
Translate to Upper Case	
Line (U LCA Command) (U)	358
Line (UPPERCAS) (U)	326
TT LCA Command (U)	357
Tutorial (U)	2
U Command (UP) (U)	324
U LCA Command (U)	358
Unbroken Vertical Bar ()	
Pattern Character (U)	372
UNDO Command (U)	323
Undo Member Changes (AUDITRL) (U) ...	68
UP Command (U)	324
UPP Command (UPPERCAS) (U)	326
UPPER Command (UPPERCAS) (U)	326
UPPERCAS Command (U)	326
Uppercase Translate	
Line (U LCA Command) (U)	358
Line (UPPERCAS) (U)	326
User	
to Assign Slave to (CHECKASN) (U)	83
to Change Password (PASSWORD) (U)	247
to Display List (LIBRARYU) (U)	205
to Set Mail Proxy (ATTACHX) (U)	64
UU LCA Command (U)	358
Value Operands - Rules for Entry (U)	18
Variable	
Overview (U)	41
View	<i>See Display</i>
VIEW Command	
Reference (U)	327
VSE	<i>See also POWER</i>
Batch Features Discussion (U)	35
Sublibrary	<i>See Sublibrary</i>
VSE.MESSAGES.ONLINE (MESSAGE) (U)	236
VTOC Command (U)	329
W LCA Command (U)	359
Wide Display	<i>See Alternate Screen</i>
Wildcard	
(?,*,@) Pattern Characters (U)	374
use in Extended Search Pattern (U)	374
Windows	<i>See Split Screen</i>
Word	
Begin (<) Pattern Character (U)	373
End (>) Pattern Character (U)	373
to Search for (U)	373
Wordwrap Text (FORMAT) (U)	146
Write Console (CONSOLEI) (U)	91
Write Line	<i>See Insert Line</i>
WW LCA Command (U)	359
X LCA Command (U)	360
Y LCA Command (U)	362
Error! Cannot open file referenced on page 397	