PERKIN-ELMER

# OS/32 EDIT

User Guide

48-008 R00

The Perkin-Elmer Corporation, Computer Systems Division 2 Crescent Place, Oceanport, New Jersey 07757

# TABLE OF CONTENTS

CHAPTERS (Continued)

CHAPTERS (Continued)

# PREFACE

This manual describes the Operating System/32 (OS/32) Edit Utility, its commands, error messages, and the basic editor concepts. This manual is especially written for the new user.

Chapter 1 introduces Edit and outlines the system requirements. Chapter 2 describes building, loading, and starting the editor from the system console and an MTM terminal. Chapter 3 describes all the Edit commands and gives examples of their use. Chapter 4 guides a user through a sample editing session.

Appendix A summarizes Edit commands. Appendix B is a message summary. Appendix C contains ASCII-EBCDIC conversion tables.

This manual replaces 29-576 R01. It is rewritten and reorganized. The following commands are added: AFTER, AGAIN, BEFORE, BOTTOM, DONE, HELP, MOVE, NOFIND, OPTION COMMAND, OPTION SCREEN, RULER, SCREEN, SUBSTITUTE, TOP, and TRUNCATE. Various changes are made to existing commands. Edit now supports full hexadecimal editing.

These publications can be used in conjunction with this manual:

| MANUAL TITLE | PUBLICATION NUMBER |
|---|---|
| Operating System/32 (OS/32) Operator Reference Manual | S29-574 |
| Operating System/32 (OS/32) Accounting Facility Reference Manual | 48-007 |
| Operating System/32 (OS/32) Link Reference Manual | 48-005 |

## N O T I C E

---

The Perkin-Elmer OS/32 Link will replace the Perkin-Elmer OS/32 Task Establisher Task (TET). Link is supported by OS/32 R05.2 and higher.

For current users of TET, see the OS/32 Link Reference Manual for a comparison of Link and TET commands and a TET command description.

---

# CHAPTER 1
# OPERATING SYSTEM/32 (OS/32) EDIT

## 1.1 INTRODUCTION

OS/32 Edit is a disc-based editor that operates on an OS/32 system and on an OS/32 system with MTM. Edit can be executed in an interactive or batch environment and editing is performed on a line-by-line basis. Data can be appended to a file, altered, and saved via the commands described in Chapter 3.

## 1.2 EDIT REQUIREMENTS

These system resources are required for Edit:

- The minimum hardware required for OS/32
- Disc devices
- Temporary file support

The pure segment of the editor requires 32.50kb of memory. Each impure segment requires a minimum of 9.25kb of memory. This minimum amount includes 4kb for a buffer in which the editor maintains tables.

## 1.3 SYSTEM ENVIRONMENTS

Edit can be executed in an interactive or batch environment which is determined by the user-specified command input device in the START command. See section 2.4.

### 1.3.1 Interactive Environment

In an interactive environment, a dialogue is carried on between the user and the system. Edit waits for your command, processes it, then prompts you for the next command. Commands are entered interactively through such devices as a CRT or teletype. There are two modes of operation in an interactive environment:

- Command mode
- Input mode

During command mode, editing is performed on a copy of your file residing in memory. You can change, alter, delete, and locate data that has been previously contained in the file.

During input mode, you can change existing data and enter new data in a file. Entering a null line (a carriage return being the only new information on an input line) transfers you from input mode to command mode.

### 1.3.1.1 Prompts

In command mode, a greater than sign (>) is displayed to the list device. It is a command prompt informing the user to enter a command:

```
    >
```

When a command is typed in, followed by a carriage return, Edit interprets the command and performs the operation. Some commands are repeated every time the carriage return is depressed until a new command is entered.

Input mode causes a line prompt to be displayed. The prompt character is preceded by a line number:

```
    1   >
    2   >
```

In input mode, Edit will continue to display line prompts until a null line is entered. Edit will then return to command mode.

### 1.3.1.2 Control Characters

Some interactive devices allow you to modify data before it is transmitted to the buffer via control characters. Table 1-1 lists the control characters and their functions.

TABLE 1-1 EDIT CONTROL CHARACTERS

| CHARACTER | FUNCTION | DEVICE |
|---|---|---|
| CTRL X | Voids line being typed | CRT, TTY, MODEL 1100, 1200 VDU, Carousel |
| <-- | Deletes character directly preceding symbol | CRT, TTY, Carousel |
| CTRL H | Moves cursor back 1 space, deleting character in that space | CRT, MODEL 1100, 1200 VDU, Carousel |

### 1.3.2 Batch Environment

In a batch environment, a sequence of Edit commands is input from a disc file, card reader, or magnetic tape device and is processed as a background task. The list device is specified in the START command. User commands, Edit responses and messages are output to the list device.

If an error occurs during batch processing, execution of the current job is halted and this message is displayed:

```
-END OF TASK CODE = 2
```

Normal completion is indicated by the following message:

```
-END OF TASK CODE = 0
```

## 1.4 FILES

Data is retained by saving it to a file residing on a device. Files are identified by a unique name and specified by file descriptors according to the standard OS/32 format.

### 1.4.1 File Descriptors

File descriptors, abbreviated as fd, are entered in a standard format.

**Format:**

$$\begin{bmatrix} \text{voln:} \\ \text{dev:} \end{bmatrix} \text{ [filename] [.[ext]] [/ file class]}$$

**Parameters:**

| | |
|---|---|
| voln: | is a 1- to 4-character alphanumeric string specifying the name of a disc volume. The first character must be alphabetic and the remaining, alphanumeric. If the volume name is omitted, the default is the system or user volume. |
| dev: | is a 1- to 4-character alphanumeric string specifying a device name. The first character must be alphabetic and the remaining, alphanumeric. |

| | |
|---|---|
| filename | is a 1- to 8-character alphanumeric string specifying the name of a file. The first character must be alphabetic and the remaining, alphanumeric. If a filename is specified when a device name is specified, the filename is ignored. |
| .ext | is a 1- to 3-character alphanumeric string specifying the extension to a filename. |
| file class | is a 1-character alphabetic string specifying the type of file class. The file class types are:<br><br>P for private file<br><br>G for group file<br><br>S for system file<br><br>If the file class is omitted, the default is P in an MTM environment, and S in an operating system environment. |

## 1.5 EDIT FEATURES

### 1.5.1 ASCII and Hexadecimal Editing

Edit provides two modes of editing:

- ASCII mode
- Hexadecimal (hex) mode

In ASCII mode, Edit interprets character strings as they are represented. In hex mode, Edit interprets character strings as pairs of hexadecimal digits. All editor commands can be used in hex as well as ASCII mode. Choice of modes is specified in the OPTION command. ASCII-EBCDIC conversion tables are in Appendix C.

### 1.5.2 Identifying Data

#### 1.5.2.1 Line Numbers

Line numbers provide a convenient way to identify lines of data during an editing session. Edit assigns line numbers beginning with 1, incremented by one, up to line number 99999. The APPEND command assigns line numbers in increments of one, and is used to enter new data. The INSERT, REPLACE, and INCLUDE commands insert new information into existing data at a specified position. Line numbers for inserted lines have a decimal part following the integer number.

The decimal part is generated by incrementing the existing line number by .01. The line number of each inserted line is .01 greater than the previous line number. Incrementing stops if a new line number is the same as an existing line number. Thus, several lines can exist with the same line number. In this situation, if a line number parameter is used to specify a line, the first line with the duplicate number following the current line is assumed. Other lines can be located by using the plus (+) or minus (-) commands to position the desired lines. Enter a SAVE command to renumber the entire file.

### 1.5.2.2 Character Strings

A line of data can be identified by specifying a sequence of unique characters within the line. When a string is specified in a command format, the string identifies a line to the editor. When the specified string is not unique to one line being edited, the first line containing the string is assumed to be the desired line.

Format:

```
/string/[col]
```

Parameters:

| | |
|---|---|
| / | is a non-alphanumeric character delimiting a string. See section 1.5.2.4. |
| string | is a unique, alphanumeric sequence of characters. |
| col | is a decimal number from 1 to 256 specifying the starting column location of a character string. |

### 1.5.2.3 Current Line

During editing, the editor maintains a line pointer positioned at the current line. Edit commands with no range parameter specified operate on the current line. When changing from input mode to command mode, the current line is the last line entered from the command device; when changing from command mode to input mode, the current line is the line following the last line edited.

### 1.5.2.4 Delimiters

A delimiter is a non-alphanumeric character that separates character strings in an input line. Delimiters allow the editor to recognize the beginning and end of a string. Any non-alphanumeric character is valid except:

- semicolon
- comma
- minus sign
- blank
- a character that appears in the string itself

The slash (/) is used to represent a delimiter in this manual.

### 1.5.2.5 Ranges

A range refers to specific line numbers in a file, or unique character strings that identify specific line numbers in a file. A minus sign is used to separate range arguments.

Format:

$$\left[\left\{\begin{matrix} \text{line no}_1 \\ /\text{string}_1/[\text{col}] \end{matrix}\right\}\right] \left[ - \left[\left\{\begin{matrix} \text{line no}_n \\ /\text{string}_2/[\text{col}] \end{matrix}\right\}\right]\right]$$

Parameters:

| | |
|---|---|
| line $\text{no}_1$ | is the line or beginning line of a range to be operated on. |
| line $\text{no}_n$ | is the ending line of the range to be operated on. |
| / | is a non-alphanumeric character delimiting a string. |
| $\text{string}_1$ | is a unique, alphanumeric character string or beginning string of a range to be operated on. |
| $\text{string}_2$ | is a unique, alphanumeric character string or ending string of a range to be operated on. |
| col | is a decimal number from 1 to 256 indicating the column locations of $\text{string}_1$ and $\text{string}_2$. |

Functional Details:

Any combination of range arguments is permitted. Edit searches the file for the specified lines or strings. After a match is made, the actual function of the command is performed.

Data is operated on beginning with the first line or string specified, and continuing through the last line or string specified. If the second argument is omitted, all lines from the line or string specified to the end-of-file are operated on. If the first argument is omitted, all lines from the current line to the line or string specified are operated on.

If the entire range parameter is omitted, the default is the current line.

**Examples:**

| | |
|---|---|
| 1-16 | indicates lines 1 through 16 inclusive. |
| range parameter omitted | specifies current line only. An exception to this is the SAVE command where, if the range is null, all lines of data are still saved. |
| 1 | indicates line 1 only. |
| -10 | specifies current line through line 10 inclusive. |
| - | specifies current line through end-of-file inclusive. |
| 1- | indicates line 1 through end-of-file. |
| /CHARS/-5 | is the line where the character string CHARS appears to line 5. |
| /CHARS/-/DELN/ | are the lines identified by the first occurrence of CHARS to the first occurrence of DELN. |

## 1.5.2.6 Carriage Return

The carriage return transmits a line of data to be processed and switches an Edit user from input mode to command mode.

### 1.5.3 Formatting Data

Edit presets options that control the data format of a file. These defaults allow the user to create a file without having to specify line length, line numbers, etc. See section 3.19.

#### 1.5.3.1 Line Length

The maximum number of characters allowed in each line is called line length. When appending to a new file i.e. no GET command is issued prior to the APPEND, the record length defaults to 80 bytes. If a GET is issued to an existing disc file and OPTION LENGTH (see OPTION command) was not specified, the length defaults to the record length of the file. The maximum characters per line is 256. You can specify line length via the length parameter in the OPTION command.

If the number of bytes per record is less than the editor line length, each line is padded with blanks. You can specify a column number, via the TRUNCATE command, indicating the column at which truncation will occur. Characters displayed beyond the specified column are lost when a line of data is expanded. The default is column 73.

#### 1.5.3.2 Text Display

Several commands cause data to be displayed with the appropriate line number. The decimal portion of the integer is only output if it is not equal to zero. Trailing blanks are not output. Data is continued on the next line, up to 80 characters, when necessary.

When the tab option is in effect, data is displayed according to column settings defined by the tab option. See section 1.5.3.3. When the mode specified in the OPTION command is hexadecimal, all data is displayed in hexadecimal format.

#### 1.5.3.3 Tab Settings

Tab control causes data tc be aligned at preset positions and frees the user from manually spacing data.

The tab parameter of the CPTION command allows the user to set tabs and designate a tab control character. The tab settings are used only when text is entered at the command device. The editor expands the line entered according to the tab control character and settings, and retains the line in its expanded form. See section 3.19.

#### 1.5.3.4 Column Display

Edit displays a ruler at the user's console via the RULER command. Use this as a guide to determine where columns begin and end.

Example:

```
>RULER

   0         1         2         3         4         5         6
   1234567890123456789012345678901234567890123456789012345678901234567
```

When hexadecimal is entered as the mode in the OPTION command, column numbers are output in hex. For example:

```
>RULER

   0102030405060708090A0B0C0D0E0F101112131415161718191A1B1C1D1E1F202122
```

## 1.6 VERIFICATION

Lines that are located, changed, or altered are automatically displayed to the list device. Editing verification can be turned off via the noverify option in the OPTION command.

## 1.7 COMMANDS

Commands are entered following the command prompt and are transmitted for processing by entering a carriage return.

### 1.7.1 Edit Command Syntax

Multiple commands can be entered on one line if they are separated by semicolons (;). An exception to this is the AGAIN command which must appear by itself on a command line. When multiple commands are entered on the same line, they are executed sequentially. If an error occurs, any subsequent commands on the line are ignored.

In a batch environment parameters can be continued by entering a comma as the last character and continuing the parameters on the following line.

### 1.7.2 Statement Syntax Conventions

These statement syntax conventions are used in all command and instruction formats:

| | |
|---|---|
| Capital letters, parentheses, and punctuation marks | must be entered exactly as shown |

| | |
|---|---|
| Lowercase letters<br>n | represent parameters or information provided by the user |
| Underlining<br>$\underline{P}$AUSE | indicates only the underlined portion of the entry is required |
| Ellipsis<br>...<br>$\text{param}_1,\ldots,\text{param}_5$ | represents an indefinite number of parameters or a range of parameters |
| Lettering with shading<br>n | represents a default option |
| Braces<br>{ } | represent required parameters from which one must be chosen |
| Brackets<br>[ ] | represent an optional parameter that can be chosen |
| Commas<br>, | separate parameters and substitute missing positional parameters |
| Braces inside brackets<br>[{ }] | represent optional parameters from which one can be chosen |
| Comma preceding braces inside brackets<br>[,{ }] | must be entered if one of the optional parameters is chosen However, if the parameters are not positional and the first parameter of the statement is not chosen, the parameter specified as the first is not preceded by a comma. |

| | |
|---|---|
| Comma inside brackets<br>[,] | must be entered if the optional parameter is chosen |
| Comma outside brackets except last parameter<br>[ ], [ ], [ ] [,] | must be entered in place of missing positional parameters and to separate optional parameters that are chosen. Commas are omitted for trailing parameters and a comma must be entered with the last specified parameter. |
| Equal sign separating keyword from parameters<br>KEYWCRD=param | must be entered to associate parameter with keyword |

## 1.8 MESSAGES

Four types of messages are output to the command device during an Edit session:

- Messages pertaining to incorrect syntax
- Messages pertaining to incomplete execution of a command
- Messages pertaining to incomplete command execution because of a more serious error
- Information messages

Messages indicating incorrect syntax are preceded by a question mark (?) and are generated when the editor cannot recognize a command or its parameters. The editor does not attempt to execute the specified command.

Execution messages are preceded by an exclamation point (!) and are output during executicn. Determine where in the file execution stopped, correct the error, and re-execute the command.

Messages preceded by two exclamation points (!!) denote a serious problem. In scme cases, the only way to recover is to attempt to save the current data to a file different from the source file, end the session, and restart.

Infcrmation messages are cutput to help you in an editing session. Some examples of these messages are reminders to save text and indications that a function has completed normally.

For a complete list of messages and meanings see Appendix B.

# CHAPTER 2
# BUILDING, LOADING, AND STARTING THE EDITOR

## 2.1 BUILDING EDIT AS A SYSTEM IMAGE LOAD MODULE

This sequence of commands builds the shared HELP segment and Edit task as a system image load module:

```
LOAD LINK
START
ESTABLISH SHARED,ADDR=E0000        Build the HELP segment
INCLUDE HELP
TITLE
MAP PR:,ADDR,ALPHA
BUILD HELP

ESTABLISH TASK                     Build the Edit task
OPTION SEG,WORK=100,SYS=FFFFF
SHARED HELP
INCLUDE EDIT
MAP PR:,ADDR,ALPHA
BUILD EDIT
END
```

For an explanation of these commands, see the Operating System/32 (OS/32) Link Reference Manual.

## 2.2 LOADING EDIT FROM AN MTM TERMINAL

After building the image load module, EDIT.TSK, the module can be loaded into memory with this command:

**Format:**

```
LOAD fd[,segsize increment]
```

**Parameters:**

| | |
|---|---|
| fd | is the file descriptor of the Edit load module to be loaded into memory. |

| | |
|---|---|
| segsize increment | is a decimal number in kb specifying the additional memory to be added to the task's impure segment. |

Functional Details:

When the Edit load module is loaded into memory, the system displays an asterisk (*) prompt, indicating the editor can be started.

In order to successfully load the task, the shared HELP segment must be on the volume under the account number from which the task is to be loaded, or on the default system volume under account number 0.

## 2.3 LOADING EDIT FROM THE SYSTEM CONSOLE

The LOAD command loads Edit from the system console.

Format:

```
    LOAD taskid[,[fd][,segsize increment]]
```

Parameters:

| | |
|---|---|
| taskid | specifies the name of the module after it is loaded into the foreground segment of memory as the currently-selected task. |
| fd | is the file descriptor of the editor image load module to be loaded into memory. If this parameter is omitted, the default is taskid.TSK. |
| segsize increment | is a decimal number in kb specifying the additional memory to be added to the task's impure segment. |

Functional Details:

The system sets Edit as the currently-selected task. When the system displays the asterisk prompt, the editor can be started.

## 2.4 STARTING EDIT

The START command starts Edit.

Format:

START [,COMMAND=$fd_1$] [,LIST=$fd_2$]

Parameters:

| | |
|---|---|
| COMMAND= | $fd_1$ specifies the input device from which commands are to be entered. If this parameter is omitted, the default is the console device (CON:). If the specified command input device is interactive and the LIST parameter is omitted, all output data and messages are sent to the command device. If the command input device is batch, the LIST parameter must be specified. |
| LIST= | $fd_2$ specifies the output device to which output data is sent. If this parameter is omitted and the command device is interactive, the output data and all messages are sent to the command device. If the command device is batch, the LIST parameter must be specified. |

Functional Details:

If both parameters in the START command are omitted, the default is the console device (CON:).

Error Messages:

!INVALID COMMAND DEVICE

The file or device specified as the command device of the START command is syntactically incorrect.

!INVALID LIST DEVICE

The file or device specified as the list device is syntactically invalid.

!DUPLICATE START OPTION

A command or list parameter was entered more than once.

!SYNTAX ERROR IN START OPTIONS

The command or list parameter was entered incorrectly.

!LIST OPTION OMITTED - BATCH MODE

A non-interactive device was specified as the command device, and the list parameter was not specified.

!UNABLE TO ASSIGN LIST DEVICE

The file or device specified as the list device cannot be assigned.

-END OF TASK CODE=1

The editor cannot proceed without valid START parameters. Code 1 indicates abnormal job termination.

# CHAPTER 3
# OPERATING SYSTEM/32 (OS/32) EDIT COMMANDS

## 3.1 INTRODUCTION

This chapter describes the following OS/32 Edit commands:

- AFTER
- AGAIN
- APPEND
- BEFORE
- BOTTOM
- CHANGE
- COLUMN
- DELETE
- DONE
- END
- FIND
- GET
- HELP
- INCLUDE
- INSERT
- MOVE
- NOFIND
- OPTION
- PAUSE
- REPLACE
- RULER
- SAVE
- SCREEN
- SEND STOP
- SUBSTITUTE
- TOP
- TRUNCATE
- TYPE
- (+) PLUS SIGN
- (-) MINUS SIGN
- LINE NUMBER

## 3.2 AFTER COMMAND

The AFTER command inserts a new character string immediately after a user-specified character string in a line of data.

Format:

$$\underline{\text{AF}}\text{TER /string/newstring}\left[/\left[\begin{Bmatrix}\text{line no}_1 \\ \text{/string}_1\text{/ [col]}\end{Bmatrix}\right]\left[-\left[\begin{Bmatrix}\text{line no}_n \\ \text{/string}_2\text{/ [col]}\end{Bmatrix}\right]\right]\right]$$

Parameters:

| | |
|---|---|
| / | is a non-alphanumeric character delimiting a string. The blank, minus sign, comma, semicolon, and any character that appears in the string itself cannot be used. |
| string | is a sequence of unique alphanumeric characters. The new string is inserted after this string. |
| col | is a decimal number from 1 to 256 indicating the starting column location of $\text{string}_1$ and $\text{string}_2$. |
| newstring | is a sequence of unique alphanumeric characters to be inserted after the specified character string. |
| line $\text{no}_1$ | is the line number or first line number of a range of lines where the newstring will be inserted after the specified string. |
| line $\text{no}_n$ | is the ending line number of a range where the newstring will be inserted after the specified string. |
| $\text{string}_1$ | is a unique character string or first unique character string of a range which determines where in the file the newstring will be inserted after the specified string. |

| | |
|---|---|
| $string_2$ | is the ending unique character string of a range which determines where in the file the newstring will be inserted after the specified string. |

**Functional Details:**

If the inserted string produces a line longer than the established line length, or if the portion of data to the left of the truncate column is expanded, the following message is displayed when any non-blank characters are lost:

```
!NON-BLANK CHARACTERS LOST
```

**Examples:**

```
>AFTER/DISC/FILE/2
```

After the character string DISC, insert character string FILE in line 2.

```
>AFTER/DISC/FILE/2-/TAPE/
```

After the character string DISC, insert character string FILE from line 2 to the first occurrence of the character string TAPE.

```
>AFTER/DISC/FILE/2-
```

After the character string DISC, insert the character string FILE from line 2 to end-of-file.

## 3.3 AGAIN COMMAND

The AGAIN command re-executes a previously specified command line.

**Format:**

    AGAIN

**Functional Details:**

Commands that are not re-executed are DELETE, GET, END, and a line of data entered with a line number. When the AGAIN command is entered following these commands, the following message is displayed:

```
?REPEAT DISALLOWED
```

The AGAIN command must be the only command on the input line.

**Examples:**

```
>TY 1-4
    1  This is an example
    2  of how
    3  the AGAIN
    4  command works.
>AG
    1  This is an example
    2  of how
    3  the AGAIN
    4  command works.
```

## 3.4 APPEND COMMAND

The APPEND command enters data to a file from the command input device.

**Format:**

```
    APPEND
```

**Functional Details:**

Enter APPEND to either create a new file or add data to an existing file.

In an interactive environment, Edit prompts you for data by displaying line numbers. The first line of data entered is assigned line number 1. Subsequent line numbers are incremented by 1. The maximum lines allowed in a file is 99999. If data already exists, the new information is entered on the line following the last line of data. Tab settings are recognized, if set during the current editing session.

In a batch environment, data is entered immediately following the APPEND command. Terminate data with the character string previously specified in the OPTION command. See section 3.19.

**Examples:**

```
>AP
    1   >To create
    2   >a new file
    3   >enter APPEND.
    4   >
>TY 1-
    1    To create
    2    a new file
    3    enter APPEND.
```

## 3.5 BEFORE COMMAND

The BEFORE command inserts a new character string immediately before a user-specified character string in a line of data.

Format:

$$\underline{\text{BE}}\text{FORE /string/newstring}\left[ / \left[ \begin{Bmatrix} \text{line no}_1 \\ \text{/string}_1\text{/[col]} \end{Bmatrix} \right] \left[ - \left[ \begin{Bmatrix} \text{line no}_n \\ \text{/string}_2\text{/[col]} \end{Bmatrix} \right] \right] \right]$$

Parameters:

| | |
|---|---|
| / | is a non-alphanumeric character delimiting a string. The blank, minus sign, comma, semicolon, and any character that appears in the string itself cannot be used. |
| string | is a sequence of unique alphanumeric characters. The new string is inserted before this string. |
| col | is a decimal number from 1 to 256 indicating the starting column location of $\text{string}_1$ and $\text{string}_2$. |
| newstring | is a sequence of unique alphanumeric characters to be inserted before the specified character string. |
| line $\text{no}_1$ | is the line number or first line number of a range of lines where the newstring will be inserted before the specified string. |
| line $\text{no}_n$ | is the ending line number of a range where the newstring will be inserted before the specified string. |
| $\text{string}_1$ | is a unique character string or first unique character string of a range that determines where in the file the newstring will be inserted before the specified string. |

| | |
|---|---|
| $\text{string}_2$ | is the ending unique character string of a range that determines where in the file the newstring will be inserted before the specified string. |

Functional Details:

If the inserted string produces a line longer than the established line length, or the portion of text to the left of the truncate column is expanded, the following message is displayed if any non-blank characters are lost:

```
!NON-BLANK CHARACTERS LOST
```

**Examples:**

```
>BEFORE /DISC/FILE/2
```

Before the character string DISC, insert the character string FILE in line 2.

```
>BEFORE /DISC/FILE/2-/TAPE/
```

Before the character string DISC, insert the character string FILE from line 2 to the first occurrence of character string TAPE.

```
>BEFORE /DISC/FILE/2-
```

Before the character string DISC, insert the character string FILE from line 2 to end-of-file.

## 3.6 BOTTOM COMMAND

The BOTTOM command changes the current line pointer to the last line of the file.

Format:

```
    BOTTOM
```

Examples:

```
    >TY 1-
        1    The BOTTOM command
        2    changes the current line pointer
        3    to the last line of the file.
    >TY 1
        1    The BOTTOM command
    >BOTTOM
        3    to the last line of the file.
```

## 3.7 CHANGE COMMAND

The CHANGE command replaces a specified character string with new data.

Format:

$$\text{\underline{CH}ANGE /oldstring/[col],/newstring/}$$

$$\left[ , \left\{ \begin{array}{l} \left\{\begin{array}{l}\text{line no}\\ \text{/string/[col]}\end{array}\right\} \\ \left[\left\{\begin{array}{l}\text{line no}_1\\ \text{/string}_1\text{/[col]}\end{array}\right\}\right] - \left[\left\{\begin{array}{l}\text{line no}_n\\ \text{/string}_2\text{/[col]}\end{array}\right\}\right]\end{array}\right\}\right]$$

Parameters:

| | |
|---|---|
| / | is a ncn-alphanumeric character delimiting a string. The comma, minus sign, blank, semicolon, and any character in the string itself cannot be used. |
| oldstring | is a unique character string to be replaced by the newstring. |
| col | is a decimal number from 1 to 256 indicating the starting column location of the character string tc be changed. |
| newstring | is a unique character string which replaces the oldstring. |
| line $no_1$ | is a line number or first line number of a range to be searched for the oldstring. |
| line $no_n$ | is the ending line number of a range to be searched for the oldstring. |
| $string_1$ | is a unique character string or first string of a range that determines where in the file the oldstring will be replaced by the newstring. |
| $string_2$ | is the ending unique character string of the |

range that determines where in the file the oldstring will be replaced by the newstring.

col — is a decimal number from 1 to 256 indicating the starting column location of $string_1$ and $string_2$.

Functional Details:

When CHANGE is used, all occurrences of the oldstring are replaced by the newstring.

If a tab character is contained within a string, it is treated the same as any character in the string.

If the inserted string causes the line length to exceed the default, or the portion of data to the left of the truncate column is expanded, the following message is displayed if non-blank characters are lost:

```
    !NON-BLANK CHARACTERS LOST
```

Examples:

```
    >CHANGE.READ.,.INPUT.,3-
```

Change all occurrences of READ to INPUT, from line 3 through end-of-file.

```
    >CHANGE?READ?,?INPUT?,1-20
```

Change all occurrences of READ to INPUT, from line 1 through line 20.

```
    >CHANGE.READ.,. .,1-
```

Change all occurrences of READ to a space, from line 1 to end-of-file.

## 3.8 COLUMN COMMAND

The COLUMN command replaces data beginning at a specified column with a new character string.

Format:

$$\underline{\text{CO}}\text{LUMN col no,/string/}\left[,\left\{\begin{array}{l}\left\{\begin{array}{l}\text{line no}\\ \text{/string/[col]}\end{array}\right\}\\ \left[\left\{\begin{array}{l}\text{line no}_1\\ \text{/string}_1\text{/[col]}\end{array}\right\}\right]-\left[\left\{\begin{array}{l}\text{line no}_n\\ \text{/string}_2\text{/[col]}\end{array}\right\}\right]\end{array}\right\}\right]$$

Parameters:

| | |
|---|---|
| col no | is a decimal number from 1 to 256 specifying the beginning column where data is to be replaced. |
| / | is a non-alphanumeric character delimiting a string. The comma, blank, minus sign, semicolon, and any character that appears in the string itself cannot be used. |
| string | is a sequence of alphanumeric characters, specified by delimiters, that replaces the data beginning at the specified column. |
| line $no_1$ | is the line number or first line number of a range of lines where data will be replaced at the specified column. |
| line $no_n$ | is the ending line number of a range where data will be replaced at the specified column. |
| $string_1$ | is a unique character string or the first string of a range that determines where data will be replaced in a file. |
| $string_2$ | is the ending unique character string of a range that determines where data will be replaced in a file. |
| col | is a decimal number from 1 to 256 indicating |

the starting column location of $string_1$ and $string_2$.

Functional Details:

If the inserted string produces a line longer than the established line length, or the portion of text to the left of the truncate column is expanded, the following message is displayed if non-blank characters were lost:

```
!NON-BLANK CHARACTERS LOST
```

**Examples:**

```
>COL 16,/R3/
```

Change the data starting in column 16 to the character string R3 in the current line.

```
>COL 10,/LH/,1-2
```

Change the data starting in column 10 to the character string LH, in lines 1 through 20.

```
>COL 15,/A2B/,/A1B/-/A5B/
```

Change the data starting in column 15 to the character string A2B, from the line containing the character string A1B to the line containing the character string A5B.

```
>COL 5,/DISC/,-/FILE/
```

Change the data starting in column 5 to the character string DISC, from the current line to the line containing the character string FILE.

```
>COL 5,/DISC/,-
```

Change the data starting in column 5 to the character string DISC from the current line to the end-of-file.

## 3.9 DELETE COMMAND

The DELETE command deletes data from a file.

Format:

$$\underline{\text{DE}}\text{LETE}\left[\begin{Bmatrix}\text{line no}_1\\ /\text{string}_1/[\text{col}]\end{Bmatrix}\right]\left[-\left[\begin{Bmatrix}\text{line no}_n\\ /\text{string}_2/[\text{col}]\end{Bmatrix}\right]\right]$$

Parameters:

| | |
|---|---|
| line $\text{no}_1$ | is the line number or first line number of a range to be deleted. |
| line $\text{no}_n$ | is the ending line number of a range of data to be deleted. |
| / | is a non-alphanumeric character delimiting a string. The comma, blank, minus sign, semicolon, and any character appearing in the string itself cannot be used. |
| $\text{string}_1$ | is a unique character string or first unique character string of a range that determines where data will be deleted in a file. |
| $\text{string}_2$ | is the ending unique character string of a range that determines where data will be deleted in a file. |
| col | is a decimal number from 1 to 256 indicating the column location of $\text{string}_1$ and $\text{string}_2$. |

Functional Details:

If all lines of data are deleted the following message is displayed:

```
ALL LINES DELETED
```

The DELETE command cannot be re-executed by entering the AGAIN command.

Examples:

```
>DELETE 10-20
```

Delete lines 10 through 20,

```
>DELETE
```

Delete the current line.

```
>DELETE-
```

Delete all data from the current line to the end-of-file.

```
>DELETE 20-
```

Delete line 20 through the end-of-file.

```
>DELETE /LPR/-/ACC/
```

Delete all lines, from character string LPR to character string ACC.

```
>DELETE /LPR/2-10
```

Delete all lines, from character string LPR beginning in column 2 through line 10.

## 3.10 DONE COMMAND

The DONE command saves the current file and ends the editing session.

**Format:**

```
    DONE
```

**Functional Details:**

This command is equivalent to a SAVE* followed by an END command.

## 3.11 END COMMAND

The END command ends the editing session and returns control to the operating system.

**Format:**

```
    END
```

**Functional Details:**

Edit goes to end-of-task with a code of 0 to indicate normal termination.

If data was not saved before the END command is entered, the following command is displayed:

```
    REMINDER - SAVE YOUR CURRENT TEXT
```

The END command cannot be re-executed by entering the AGAIN command.

## 3.12 FIND COMMAND

The FIND command searches for all occurences of a specified string in a given range of a file and displays them to the list device.

Format:

$$\underline{F}\text{IND /string/[col]}\left[,\left\{\begin{array}{l}\left\{\begin{array}{l}\text{line no}\\ \text{/string/[col]}\end{array}\right\}\\ \left[\left\{\begin{array}{l}\text{line no}_1\\ \text{/string}_1\text{/[col]}\end{array}\right\}\right]-\left[\left\{\begin{array}{l}\text{line no}_n\\ \text{/string}_2\text{/[col]}\end{array}\right\}\right]\end{array}\right\}\right].$$

Parameters:

| | |
|---|---|
| / | is a non-alphanumeric character delimiting a string. The comma, blank, minus sign, semicolon, and any character appearing in the string itself cannot be used. |
| string | is a sequence of alphanumeric characters specified by delimiters, to be searched for. |
| col | is a decimal number from 1 to 256 indicating the starting column location of the specified string. |
| line $no_1$ | is the line or first line number of a range of lines to be searched for all occurrences of the specified character string. |
| line $no_2$ | is the ending line number of a range to be searched for all occurrences of the specified character string. |
| $string_1$ | is a unique character string or first unique character string of a range that determines where in the file to search for the specified character string. |
| $string_2$ | is the ending unique character string of a range that determines where in the file to search for the specified character string. |

col            is a decimal number from 1 to 256 indicating the starting column location of string$_1$ and string$_2$.

Functional Details:

If a tab character is contained within a string, it is treated as just another character in that particular string.

Examples:

```
>FIND/SCH/,1-
```

Find every occurrence of the character string SCH, begining in line 1 through end-of-file.

```
>FIND/SCH/1,5-7
```

Find every occurrence of the character string SCH, beginning in column 1 from lines 5 through 7.

```
>FIND /SCH/,/ACC/2-/CCA/2
```

Find every occurrence of the character string SCH, beginning at the location of string ACC in column 2 and ending at the location of string CCA in column 2.

```
>FIND /SCH/,-/CCA/2
```

Find every occurrence of character string SCH, beginning at the current line and ending at the location of string CCA in column 2.

## 3.13 GET COMMAND

The GET command brings a file off disc and readies it for editing.

Format:

```
    GET fd
```

Parameters:

| | |
|---|---|
| fd | is the file descriptor of the file being accessed. |

Functional Details:

When a GET command is entered, a copy of the specified file is brought off disc. It is now opened for editing and data can be added, deleted, inserted, changed, etc. Not until a SAVE command is entered, does the updated version of the file replace the original version. If a GET command is entered before the current data is saved, a reminder tc save your current text is displayed. The GET command is not re-executed by entering the AGAIN command.

If the record length of the fd is different from the current line length, the lines of data are expanded or decreased, accordingly, when they are output. When records are expanded, they are padded with blanks for ASCII mode. The pad character in hex mode is X'00'.

If OPTION LENGTH was issued prior to the GET command, and the physical length of the input file is larger than the specified line length, the following message is displayed:

```
    FILE RECORD LENGTH=m, CURRENT LINE LENGTH=n
```

The variable m is the reccrd length of the input file that was established when the GET command was issued. The variable n is the line length specified by the user via the OPTION command.

In a batch environment, the program terminates. In an interactive environment, the message is informative only.

When the editor is executed as a foreground task, only files with account number 0 can be accessed.

**Examples:**

```
>GET CH1EDT.008

     Get file CH1EDT.008 on the default user volume.
```

## 3.14 HELP COMMAND

The HELP command prints out information on how to use Edit.

**Format:**

$$\underline{H}\text{ELP}\left[\begin{Bmatrix}\text{mnemonic}\\ *\end{Bmatrix}\right]$$

**Parameters:**

| | |
|---|---|
| mnemonic | is any valid Edit command mnemonic. Information will be displayed on how to use the command associated with the mnemonic specified. |
| * | causes a list of all Edit commands to be displayed. |

**Functional Details:**

The HELP file must be on the default system volume under account 0.

If parameters are omitted, Edit prints out information on how to use the HELP command.

**Examples:**

```
>HELP CH
```

Displays to the user information on the CHANGE command.

```
>HELP INS
```

Displays to the user information on the INSERT command.

## 3.15 INCLUDE COMMAND

The INCLUDE command copies a line or range of lines of data to a specified place in a file.

Format:

$$\underline{\text{INC}}\text{LUDE}\left[\begin{Bmatrix}\text{line no}\\ \text{/string/[col]}\end{Bmatrix}\right],[\text{fd}]$$

$$\left[,\begin{Bmatrix}\begin{Bmatrix}\text{line no}\\ \text{/string/[col]}\end{Bmatrix}\\ \left[\begin{Bmatrix}\text{line no}_1\\ \text{/string}_1\text{/[col]}\end{Bmatrix}\right]-\left[\begin{Bmatrix}\text{line no}_n\\ \text{/string}_2\text{/[col]}\end{Bmatrix}\right]\end{Bmatrix}\right]$$

Parameters:

| | |
|---|---|
| line no | is the line number after which data will be included. If this parameter is omitted, the data will be included immediately following the current line. |
| string | is a unique character string which determines where in the file data will be included. |
| / | is a non-alphanumeric character delimiting a string. The comma, blank, minus sign, semicolon, and any character that appears in the string itself cannot be used. |
| fd | is the file descriptor of the file containing the lines to be included. If this parameter is omitted, the current file is the default. |
| line $no_1$ | is the line number or first line number of a range of lines to be inserted after the specified line number. |
| line $no_n$ | is the ending line number number of a range of line numbers to be inserted after the specified line number. |

| | |
|---|---|
| $string_1$ | is a unique character string or first unique character string of a range that determines the lines to be inserted after the specified line number. |
| $string_2$ | is the ending unique character string of a range that determines the lines to be inserted after the specified line number. |
| col | is a decimal number from 1 to 256 indicating the starting column location of a character string. |

**Functional Details:**

If the specified range is null, and the fd is omitted, the current line is inserted. If the range is null, and fd is specified, the first line of the specified fd is inserted.

The lines are not deleted from their original place in the file. The inserted lines are assigned the line number they are inserted after, incremented by .01. Incrementing stops when the new line number equals the last existing line number. The last line number generated is then used for all subsequent lines inserted.

**Examples:**

```
>INCLUDE 2,,20-
```

Include after line 2, lines 20 through the last line of the current file.

```
>INCLUDE 2,TRY.EDT,1-
```

Include after line 2, all lines of a file named TRY.EDT.

```
>INCLUDE,,25-45
```

Include after the current line, lines 25-45 of the current file.

```
>INCLUDE,TEXT 100-200
```

Include after the current line, lines 100 through 200 of the file named TEXT.

```
>INCLUDE .END.7,TEXT,1-
```

Include, following the line containing END in column 7, all lines of the file named TEXT.

```
>INCLUDE 2,TEXT,/BEGIN/2-/END/4
```

Include, after line 2 from the file named TEXT, all lines from the line containing the character string BEGIN in column 2 to the line containing character string END in column 4.

## 3.16 INSERT COMMAND

The INSERT command inserts one or more new lines of data into a file.

Format:

$$\underline{\text{INS}}\text{ERT}\left[\left\{\begin{array}{l}\text{line no}\\ \text{/string/ [col]}\\ \text{current line}\end{array}\right\}\right]$$

Parameters:

| | |
|---|---|
| line no | is the line after which data is to be inserted. If line number is omitted, the data will be inserted after the current line. |
| / | is a non-alphanumeric character delimiting a string. The comma, blank, minus sign, semicolon, and any character in the string itself cannot be used. |
| string | is a unique character string that determines where in the file data will be inserted. |
| col | is a decimal number from 1 to 256 indicating the starting column location of the specified character string. |

Functional Details:

When the user enters the INSERT command interactively, the editor displays the next fractional line number and a prompt, to indicate that a line of data can be entered. Line numbers are displayed in increments of .01, following the line specified by the line parameter. Incrementing stops if the new line number equals the next existing line number. The last line number generated is then used for all subsequent lines inserted.

In a batch environment, input immediately follows the INSERT command. The data is terminated by a character string specified in the OPTION command.

Examples:

```
>TY 1-
    1     This is an
    2     example of
    3     INSERT command works.
>ins2
    2.01>how the
    2.02>
>TY 1-
    1     This is an
    2     example of
    2.01 how the
    3     INSERT command works.
```

## 3.17 MOVE COMMAND

The MOVE command moves a line or range of lines of data within a file.

Format:

$$\underline{\text{M}}\text{OVE}\left[\begin{Bmatrix}\text{line no}\\ \text{/string/[col]}\end{Bmatrix}\right]\left[,\begin{Bmatrix}\begin{Bmatrix}\text{line no}\\ \text{/string/[col]}\end{Bmatrix}\\ \left[\begin{Bmatrix}\text{line no}_1\\ \text{/string}_1\text{/[col]}\end{Bmatrix}\right]-\left[\begin{Bmatrix}\text{line no}_n\\ \text{/string}_2\text{/[col]}\end{Bmatrix}\right]\end{Bmatrix}\right]$$

Parameters:

| | |
|---|---|
| line no | is the line number after which lines of data are to be moved. |
| / | is a non-alphanumeric character delimiting a string. The comma, blank, minus sign, semicolon, and any character within the string itself cannot be used. |
| string | is a unique character string indicating where in the files data is to be moved. |
| line $no_1$ | is a line or first line number of a range to be moved after the specified line number. |
| line $no_n$ | is the ending line number of the range to be moved after the specified line number. |
| $string_1$ | is a unique character string or first unique character string of a range that determines the lines to be moved. |
| $string_2$ | is a unique character string or ending string of a range that determines the lines to be moved. |

| | |
|---|---|
| col | is a decimal number from 1 to 256 indicating the starting column location of a character string. |

Functional Details:

After the lines are moved, the original lines of data are deleted. The moved lines are assigned the same line number as the line they are inserted after, incremented by .01.

Examples:

```
>MOVE 4, 2-3
```

After line 4, insert lines 2 through 3. Lines 2 through 3 are deleted from their original place in the file.

```
>MOVE 4, /DISC/-/FILE/
```

After line 4, insert the lines beginning with the line containing the character string DISC to the line containing the character string FILE. The lines identified by the specified character strings are deleted from their original place in the file.

```
>TY 1-
    1    This is an
    2    example of
    3    MOVE command works.
    4    how the
>MOVE 2,4
>TY 1-
    1    This is an
    2    example of
    2.01 how the
    3    MOVE command works.
```

## 3.18 NOFIND COMMAND

The NOFIND command searches for all lines in which a specified string does not appear.

Format:

$$\underline{N}OFIND\ /string/[col]\ \left[,\left\{\begin{matrix}\left\{\begin{matrix}line\ no\\ /string/[col]\end{matrix}\right\}\\ \left[\left\{\begin{matrix}line\ no_1\\ /string_1/[col]\end{matrix}\right\}\right]-\left[\left\{\begin{matrix}line\ no_n\\ /string_2/[col]\end{matrix}\right\}\right]\end{matrix}\right\}\right]$$

Parameters:

| | |
|---|---|
| / | is a non-alphanumeric character delimiting a string. The comma, blank, minus sign, semicolon, and any character appearing in the string itself cannot be used. |
| string | is the unique alphanumeric character string to be searched for. |
| line $no_1$ | is a line number or first line number of a range to be searched for the specified character string. |
| line $no_n$ | is the ending line number of a range to be searched for the specified character string. |
| $string_1$ | is a unique character string or first unique character string of a range that determines the lines to be searched for the specified character string. |
| $string_2$ | is the ending unique character string of the range that determines the lines to be searched for the specified character string. |
| col | is a decimal number from 1 to 256 indicating the starting column location of a character string. |

Functional Details:

If the string occurs in every line specified by the line number parameter, the following message is displayed:

```
?OCCURS IN ALL LINES
```

Examples:

```
>NOFIND /DISC/,1-
```

Find all lines where character string DISC does not occur, from lines 1 to end-of-file.

```
>NOFIND /DISC/5, 1-/FILE/
```

Find all lines where character string DISC, beginning in column 5, does not occur, from line 1 to the line containing character string FILE.

```
>NOFIND /DISC/,/FILE/3-20
```

Find all lines where character string DISC does not occur, from the line containing character string FILE, beginning in column 3, through line 20.

## 3.19 OPTION COMMAND

The OPTION command sets control options for a file.

Format:

```
        ┌              ┐ ┌        ┌ ┌n┐┐┐ ┌       ┌ ┌HEXADECIMAL┐┐┐
OPTION  │COMMAND[=fd]  │ │,BLOCK  │=┤  ││ │ │,MODE  │=┤           ││
        └              ┘ └        └ └5┘┘┘ └       └ └ASCII      ┘┘┘

        ┌          ┌ ┌n ┐┐┐ ┌       ┌ ┌          fd          ┐┐┐
        │,LENGTH   │=┤  ││ │ │,LIST  │=┤                      ││
        └          └ └80┘┘┘ └       └ └fd in START command   ┘┘┘

        ┌ ┌     ┌ ┌          fd          ┐┐┐
        │,│LOG  │=┤                      │││
        │ │     └ └fd in START command   ┘┘│
        └ └NOLOG                           ┘┘

        ┌ ┌     ┌ ┌tab char,col1 [,col2,...,coln]┐┐┐
        │ │TAB  │=┤FORTRAN                       │││
        │,│     └ └CAL                           ┘┘│
        └ └NOTAB                                   ┘┘

        ┌              ┌ ┌   string    ┐┐┐ ┌         ┌ ┌n ┐┐┐
        │,TERMINATOR   │=┤     cr      ││ │ │,SCREEN  │=┤  ││
        └              └ └unspecified  ┘┘┘ └         └ └23┘┘┘

        ┌ ┌NOVERIFY┐┐
        │,┤        ││
        └ └VERIFY  ┘┘
```

Parameters:

| | |
|---|---|
| COM= | fd is a file containing a command or a sequence of commands to be executed when the fd is specified. There can be no more than 5 nested editor command files. |
| BLOCK= | n is a number from 1 to 255 which sets the data block size for allocated files. The default is 5. |
| MODE= | specifying HEXADECIMAL, sets the character format to hexadecimal. Specifying ASCII, sets the character format to ASCII. The default is ASCII. |

| | |
|---|---|
| LENGTH | n sets the line length for data input and output. This value cannot exceed 256. The default value is 80 bytes or the record length of the file. |
| LIST | fd assigns the list device to the specified fd. The default fd is the fd assigned in the START command. |
| LOG | fd assigns the log device to the specified fd. The default is the fd assigned in the START command. |
| NOLOG | cancels the log option. |
| VERIFY | causes data that was located, changed, or altered, to be output to the list device. If not specified, Verify is in effect. |
| NOVERIFY | cancels the verify option. Located, changed, or altered lines are not output to the list device. |
| TAB= | tab char is a non-alphanumeric character used to set tabs for data formatting. The tab char is followed by a decimal number from 1 to 256 specifying the columns where tabs are to be set. |
| NOTAB | clears current tab settings. |
| TERMINATOR= | is a 1- to 4-character alphabetic string used to terminate data input. Terminator can also equal a carriage return. |
| SCREEN= | n is a number from 1 to 23 that sets the number of lines per screen to be displayed. The default is 23 lines. |

**Functional Details:**

The OPTION command can be entered any time during an editing session while in command mode. Parameters can be entered on the same line, in any order, separated by commas. The editor processes them sequentially from left to right. If conflicting parameters are entered in the same statement, the last one specified is in effect. The editor displays current option settings if the command is entered with no parameters.

When the command option is specified, if an end-of-file condition occurs when reading the command file, the following message is displayed:

```
!END OF COMMAND FILE
```

There can be no more than 5 nested editor command files.

The block option sets the data block size for all subsequent file allocations. The block size is a decimal number from 1 to 255; the index block size is always 1. The value is used only when a file is allocated. At that time, if the block size specified exceeds the maximum system block size, allocation fails, and the following message is displayed:

```
!!FILE ERR, LUn, SIZE ERROR, fd
```

In the mode option, if ASCII is specified, all characters are interpreted in ASCII. If HEXADECIMAL is specified, all characters are interpreted as hexadecimal digits. Hex and ASCII modes are recognized for <u>all</u> Edit operations. When the editor is in hex mode, all data must be entered in hexadecimal. Valid hex characters are 0-9 and A-F. Special characters such as the carriage return and tab character have no effects. The pad character in hexadecimal mode is X'00'. The corresponding pad character in ASCII is ' '. The following example shows hex editing:

```
*EDIT
 >O MC=HEX
 >AP
  1    >01239FFACEDC88ACDCBB789EF55
  2    >FFFFDDDDEEEE88992136553CDACCCEDF
  3    >
 >CH /DDDD/,/0000/,2
    2  FFFF0000EEEE88992136553CDACCCEDF000000000000000000000000
       0C0C0CCC00
```

ASCII-EBCDIC conversion tables are presented in Appendix C.

When the tab option is in effect, data is expanded before it is stored. It remains in expanded form. The tab option entered with no arguments displays the current tab character and settings. Up to 20 tabulation points can be set. Once tabs are set, new tabs can still be inserted before the next highest setting. Column specification can be omitted to avoid changing both column and tab.

When TAB=FORTRAN is specified, the editor sets tabs in columns 7 and 73. Control I is the tab character.

When TAB=CAL is specified, the editor sets tabs in columns 10, 16, 36, and 73. Control I is the tab character.

The terminator option is generally used in batch environment to terminate data input. The terminator characters are positioned in columns 1-4 of the input line and are followed by blanks or a carriage return. In batch environment, when specifying a line number command to delete a line, the alternate terminator may directly follow the line number. TERMINATOR=CR specifies that a carriage return is the only valid terminator.

Examples:

```
>OPTION LENGTH=130
```

Sets the line length for data entry and output to 130 bytes per line.

```
>OPTION BLOCK=10
```

Sets the data block size to 10 for all subsequent file allocations.

```
>OPTION
```

Displays all values for current options.

```
>OPTION LI=PR:
```

Assigns the printer as the current list device.

```
>OPTION MODE
```

Displays the current mode.

```
>OPTION MODE=H
```

Sets the current mode to hexadecimal.

```
>OPTION TAB=$,5,10
```

Assigns the $ as the tab character and the tab stops at columns 5 and 10.

```
>OPTION TAB
```

Verifies the tab character and settings.

```
>OPTION TAB=S,15,TAB
```

Adds a third tab setting to those already in effect and displays the tab options.

```
>OPTION NOTAB
```

Clears the current tab settings.

```
>OPTION LOG=fd
```

Logs all commands and messages to the file or device specified by fd.

```
>OPTION NOLOG
```

Cancels the Log option. Commands and messages are not logged.

```
>OPTION TERMINATOR=END
```

Defines END, in columns 1-4, as the alternate terminator.

```
>OPTION VERIFY
```

Displays the line modified by the CHANGE command to the log device.

```
>OPTION NOVERIFY
```

Cancels the Verify option.

## 3.20 PAUSE COMMAND

The PAUSE command temporarily returns control to the operating system.

Format:

```
    PAUSE
```

Functional Details:

To return to the editor, enter the CONTINUE command. The PAUSE command does not delete or change any data in the file, and the current line is unchanged.

## 3.21 REPLACE COMMAND

The REPLACE command replaces existing lines of data with new information.

Format:

$$\underline{\text{RE}}\text{PLACE} \left[\begin{Bmatrix} \text{line no}_1 \\ /\text{string}_1/[\text{col}] \end{Bmatrix}\right] \left[-\left[\begin{Bmatrix} \text{line no}_n \\ /\text{string}_2/[\text{col}] \end{Bmatrix}\right]\right]$$

Parameters:

| | |
|---|---|
| line $\text{no}_1$ | is the line number or first line number of a range to be replaced. |
| line $\text{no}_n$ | is the ending line number of a range to be replaced. |
| / | is a non-alphanumeric character delimiting a string. The comma, blank, minus sign, semicolon, and any character appearing in the string itself cannot be used. |
| $\text{string}_1$ | is a unique character string or first unique character string of a range that determines where in the file data is to be replaced. |
| $\text{string}_2$ | is the ending unique character string that determines where in the file data is to be replaced. |
| col | is a decimal number from 1 to 255 indicating the column location of $\text{string}_1$ and $\text{string}_2$. |

Functional Details:

In an interactive environment, the editor responds to a REPLACE command with a line number as the prompt. The first number is the line specified in the command. Succeeding lines are displayed in increments of .01.

In a batch environment, input immediately follows the REPLACE command. The file is terminated by a specified string in the OPTION command.

Examples:

```
    >REPLACE 4

         Allows the user to re-enter new data on line 4.


    >TY 1-
        1    This is an
        2    example of
        3    how the
        4    MOVE command works.
    >REPLACE
        4   >REPLACF command works
        4.01>
    >TY 1-
        1    This is an
        2    example of
        3    how the
        4    REPLACE command works.
```

## 3.22 RULER COMMAND - must be in edit

The RULER command displays a ruler to the list device.

**Format:**

```
    RULER
```

**Functional Details:**

You can enter the RULER command any time during command mode of an editing session.

The ruler is output in hex, if OPTION MODE=HEX is specified in the OPTION command.

**Examples:**

```
>RULER
   0         1         2         3         4         5         6
   1234567890123456789012345678901234567890123456789012345678901234567

>OPTION MODE=H
>RULER
   0102030405060708090A0B0C0D0E0F101112131415161718191A1B1C1D1E1F202122
```

## 3.23 SAVE COMMAND

The SAVE command outputs data to a file or device.

Format:

$$\underline{\text{SA}}\text{VE}\begin{Bmatrix}\text{fd}\\ *\end{Bmatrix}\left[,\left\{\begin{matrix}\begin{Bmatrix}\text{line no}\\ /\text{string}/[\text{col}]\end{Bmatrix}\\ \left[\begin{Bmatrix}\text{line no}_1\\ /\text{string}_1/[\text{col}]\end{Bmatrix}\right]-\left[\begin{Bmatrix}\text{line no}_n\\ /\text{string}_2/[\text{col}]\end{Bmatrix}\right]\end{matrix}\right\}\right]$$

Parameters:

| | |
|---|---|
| fd | is the file descriptor of the allocated file to which the data is to be saved. |
| * | saves new data to the current input file. |
| line $\text{no}_1$ | is the line number or first line number of a range to be saved. |
| line $\text{no}_n$ | is the ending line number number of a range to be saved. |
| / | is a ncn-alphanumeric character delimiting a string. The comma, blank, minus sign, semicolon, and any character in the string itself cannot be used. |
| $\text{string}_1$ | is a unique character string or first unique character string of a range that determines the data to be saved. |
| $\text{string}_2$ | is the ending unique character string of a range that determines the data to be saved. |
| col | is a decimal number from 1 to 256 indicating the starting column location of $\text{string}_1$ and $\text{string}_2$. |

Functional Details:

If fd specifies a file that does not currently exist, an indexed file is allocated with the specified fd. If fd is the name of a file that currently exists, or is the name of the current input file, Edit displays the following message:

```
    DELETE AND REALLOCATE voln:filename/P?
```

If you indicate the file should be deleted by entering YES, Edit deletes your file. An indexed file with the specified fd is allocated, and current data is output to that file. If you indicate that the file should not be deleted by entering NO, Edit displays the following message:

```
    FILE NOT SAVED
```

The editor did not execute the SAVE and outputs a command prompt.

If a previously saved file is saved to an asterisk, or if fd is the name of the file being edited, the current data overwrites the current input file. The editor displays the following message:

```
    WORKFILE=voln:filename/P
    RENUMBERED INPUT FILE AVAILABLE,voln:filename/P
```

The data is renumbered and is available for further editing.

When the fd or range parameter is omitted, the entire file is saved. To specify the lines of data to be saved, any combination of range arguments is permitted. Edit searches the file for the specified lines or character strings. After a match is made, the actual function of the command is performed.

Data is saved beginning with the first line or string specified, continuing through the last line or string specified.

To save only the current line, explicitly specify the current line number or a unique string in that line.

Examples:

```
    >SAVE NEW.EDT
```

Save all the lines to a file named NEW.EDT on the default volume.

```
>SAVE NEW.EDT,1-10
```

Save lines 1 through 10 to a file named NEW.EDT on the default volume.

```
>SAVE M67B:NEW.EDT
```

Save all the lines to a file named NEW.EDT on volume M67B.

```
>SAVE*
```

Save the current file to the same filename.

```
>SAVE M67B:NEW.EDT,/ACC/2-/CCA/60
```

Save all data from the line identified by the first occurrence of character string ACC in column 2, to the line identified by character string CCA in column 60 to the file NEW.EDT on volume M67B.

## 3.24 SCREEN COMMAND

The SCREEN command displays a full screen of data starting, ending, or centered at the current line.

Format:

$$\underline{\text{SC}}\text{REEN}\left[\begin{Bmatrix}\text{E}\\\text{C}\\\text{S}\end{Bmatrix}\right]$$

Parameters:

| | |
|---|---|
| E | displays a full screen of data ending at the current line. If this parameter is omitted, the default is S. |
| C | displays a full screen of data centered at the current line. If this parameter is omitted, the default is S. |
| S | displays a full screen of data starting at the current line. |

```
 ----------
|   SEND   |
|   STOP   |
 ----------
```

## 3.25 SEND STOP COMMAND

The SEND STOP command allows the user to halt execution of the current command.

Format:

```
    SEND STOP
```

Functional Details:

The command is only valid during interactive use of the editor.

The SEND STOP command is recognized when Edit is executing as a foreground task, is the current task, or is a terminal task under an MTM system.

Enter SEND STOP in response to an operating system prompt. If Edit is currently outputting to the command device, depress the Break key to obtain the prompt. SEND STOP causes Edit to stop processing the command issued prior to SEND STOP. Edit is now ready to accept the next command.

## 3.26 SUBSTITUTE COMMAND

The SUBSTITUTE command replaces a specified string of data with a new string of data. In a line of data only the first occurrence of the string is replaced using this command.

**Format:**

$$\underline{\text{SU}}\text{BSTITUTE /oldstring/newstring} \left[ / \left[ \begin{Bmatrix} \text{line no}_1 \\ /\text{string}_1/ \; [\text{col}] \end{Bmatrix} \right] \left[ - \left[ \begin{Bmatrix} \text{line no}_n \\ /\text{string}_2/ \; [\text{col}] \end{Bmatrix} \right] \right] \right]$$

**Parameters:**

| | |
|---|---|
| / | is a non-alphanumeric character delimiting a string. The comma, minus sign, blank, semicolon, and any character in the string itself cannot be used. |
| oldstring | is a unique character string to be replaced by the newstring. |
| col | is a decimal number from 1 to 256 indicating the starting column location of the character string to be changed. |
| newstring | is a unique character string which replaces the oldstring. |
| line $\text{no}_1$ | is a line number or first line number of a range to be searched for the oldstring. |
| line $\text{no}_n$ | is the ending line number of a range to be searched for the oldstring. |
| $\text{string}_1$ | is a unique character string or first string of a range that determines where in the file the oldstring will be replaced by the newstring. |
| $\text{string}_2$ | is the ending unique character string of the range that determines where in the file the oldstring will be replaced by the newstring. |

| | |
|---|---|
| col | is a decimal number from 1 to 256 indicating the starting column location of $string_1$ and $string_2$. |

**Examples:**

```
>SU /SUB/READ
```

Change the first occurrence of the character string SUB to the character string READ in the current line.

```
>SU .READ..1-
```

Change the first occurrence of the character string READ to the null string from line 1 to end-of-file.

```
>SU /SUB/READ//DISC/-/FILE/
```

Change the first occurrence of character string SUB to character string READ from the line containing character string DISC to the line containing character string FILE.

```
>TY 1-
    1     A*B=B*A
    2     B*A=A*B
>SU /A/C/1-2
    1     C*B=B*A
    2     B*C=A*B
>AGAIN
    1     C*B=B*C
    2     B*C=C*B
```

## 3.27 TOP COMMAND

The TOP command changes the current line pointer to the first line of the file.

**Format:**

```
    TOP
```

**Examples:**

```
    >TY 1-
        1    The TOP command
        2    changes the current line pointer
        3    to the first line of the file.
    >TOP
        1    The TOP command
```

## 3.28 TRUNCATE COMMAND

The TRUNCATE command truncates a line of data at a specified column.

Format:

```
    TRUNCATE [col no]
```

Parameters:

| | |
|---|---|
| col no | is the column number after which data is truncated. If this parameter is omitted, the value of the current truncate column is displayed. |

Functional Details:

The specified column number becomes the point at which characters are lost when a line of data is expanded to the right of the specified column. If any non-blank characters overflow to the right of this column, they are lost, and the following message is displayed:

```
!NON-BLANK CHARACTERS LOST
```

When blank characters are truncated, no message is displayed.

## 3.29 TYPE COMMAND

The TYPE command displays lines of data to the list device.

Format:

$$\underline{\text{TY}}\text{PE} \left[\begin{Bmatrix}\text{line no}_1 \\ /\text{string}_1/[\text{col}]\end{Bmatrix}\right] \left[-\left[\begin{Bmatrix}\text{line no}_n \\ /\text{string}_2/[\text{col}]\end{Bmatrix}\right]\right]$$

Parameters:

| | |
|---|---|
| line $no_1$ | is the line or first line number of a range of lines to be displayed to the list device. |
| line $no_n$ | is the ending line number of the range to be displayed to the list device. |
| / | is a non-alphanumeric character delimiting a string. The comma, blank, minus sign, semicolon, and any character in the string itself cannot be used. |
| $string_1$ | is a unique character string or first unique character string of a range that determines what data is to be displayed. |
| $string_2$ | is the ending unique character string of a range that determines what data is to be displayed. |
| col | is a decimal number from 1 to 256 indicating the starting column of $string_1$ and $string_2$. |

Functional Details:

When a string parameter is specified, Edit begins searching from the line following the current line.

If the tab option is in effect, information is displayed appropriately.

Examples:

```
>TYPE 10-20
```

Display lines 10 through 20 of the current file.

```
>TYPE 8
```

Display line 8 of the current file.

```
>TYPE 20-
```

Display lines 20 through the last line of the current file.

```
>TYPE
```

Display the current line.

```
>TYPE -
```

Display all lines from the current line through the last line of the current file.

```
>TYPE/DISC/-/FILE/
```

Type all lines from the first occurrence of character string /DISC/ to the first occurrence of character string /FILE/.

```
>TYPE/DISC/3-/FILE/60
```

Type all lines from the first occurrence of character string /DISC/ in column 3, to the first occurrence of the character string /FILE/ in column 60.

## 3.30 PLUS SIGN COMMAND

The plus sign (+) displays an increment of the current line of a file.

**Format:**

```
+ [n]
```

**Parameters:**

| | |
|---|---|
| n | is the increment of the current line to be displayed. If this parameter is omitted, the default is the current line no+1. |

**Functional Details:**

If the line number parameter is greater than the number of lines in the file, then the last line of data is displayed along with the following message:

```
LAST LINE
```

**Examples:**

| | |
|---|---|
| >+ 2 | Display the second line following the current line. |
| >+ | Display the line following the current line. |

```
 ----------
|    -     |
 ----------
```

## 3.31 MINUS SIGN COMMAND

The minus sign (-) displays a line preceding the current line of a file.

Format:

    - [n]

Parameters:

| | |
|---|---|
| n | is the line preceding the current line to be displayed. If this parameter is omitted, the default is the current line number-1. |

Functional Details:

If the line number parameter is greater than the number of preceding lines in the text, then the first line is output, along with the following message:

```
FIRST LINE
```

Examples:

| | |
|---|---|
| >- | Display the line before the current line. |
| >-4 | Display the fourth line preceding the current line. |

## 3.32 LINE NUMBER COMMAND

Entering a line number by itself can delete, replace, or insert a line of data.

Format:

```
    n [data]
```

Parameters:

| | |
|---|---|
| n | is a line number. |
| data | is the data to be inserted or replaced. |

Functional Details:

Entering a line number followed by data, causes that data to be inserted if the line number does not exist. If the line number exists, the new data replaces the original data. If a line number is specified followed by a carriage return, that line of data is deleted. The command is not re-executed by entering the AGAIN command. In a batch environment, the terminator string specified in the OPTION command replaces the carriage return.

Examples:

| | |
|---|---|
| >2 | Delete line 2. |
| >2 ABCD | Replace data on line 2 with ABCD. Line 2 is now the current line. |
| >2.01 EFGH | Insert line 2.01 after line 2 with the data EFGH appearing on it. |
| >0.01 EFGH | Inserts a line containing character string EFGH on the line preceding the first line of a file. After the file is saved and renumbered, the line containing character string EFGH becomes the first line of the file. |

# CHAPTER 4
# SAMPLE EDIT SESSION

The following Edit sessicn was performed in an MTM environment that supports accounting.

```
>AP
    1   >The standard  includes such tigh-performance features
    2   >as capacity for up to 3  of memory, 124 gemeral-
    3   >purpose registers, mjltilevel interrupt structure,
    4   >memory relocation and protection hardware along with a
    5   >8 megabytes/second direct memory access bandwidth, dynamic
    6   >comprehensive instruction set.
    7   >a host of performance instruction sets.
    8   >To complement the 3220's inherent power, Perkin-Elmer offers
    9   >
>BEFORE /includes/processcr /1
    1   The standard processor includes such tigh-performance features
>AFTER /3/ megabytes/2
    2   as capacity for up to 3 megabytes of memory, 124 gemeral-
>MOVE 6,8
>TY 5-7
    5   8 megabytes/second direct memory access bandwidth, dynamic
    6   comprehensive instruction set.
    6.01 To complement the 3220's inherent power, Perkin-Elmer offers
    7   a host of performance instruction sets.
>SA ch4edt.008
 WORK FILE = M300:CH4EDT.00C/P
 RENUMBERED INPUT FILE AVAILABLE, M300:CH4EDT.008/P
>GET CH4EDT.008
>TY 1-
    1   The standard processor includes such tigh-performance features
    2   as capacity for up to 3 megabytes of memory, 124 gemeral-
    3   purpose registers, multilevel interrupt structure,
    4   memory relocation and protection hardware along with a
    5   8 megabytes/second direct memory access bandwidth, dynamic
    6   comprehensive instruction set.
    7   To complement the 3220's inherent power, Perkin-Elmer offers
    8   a host of performance instruction sets.

>O TA=/,29
>O TA
THE TAB  CHARACTER IS   /
   29
>AP
    9   >Cache Memory/Writable Control Store
   10   >Memory Error Logger/Floating Point Processor
   11   >Battery Backup System/Data Handling Instructions
   12   >
```

```
>TY 1-
   1    The standard processor includes such tigh-performance features
   2    as capacity for up to 3 megabytes of memory, 124 gemeral-
   3    purpose registers, multilevel interrupt structure,
   4    memory relocation and protection hardware along with a
   5    8 megabytes/second direct memory access bandwidth, dynamic
   6    comprehensive instruction set.
   7    To complement the 3220's inherent power, Perkin-Elmer offers
   8    a host of performance instruction sets.
   9    Cache Memory                  Writable Control Store
  10    Memory Error Logger           Floating Point Processor
  11    Battery Backup System         Data Handling Instructions
>TOP
   1    The standard processor includes such tigh-performance features
>BOTTOM
  11    Battery Backup System         Data Handling Instructions
>SU /tigh/high/1
   1    The standard processor includes such high-performance features
>CCL 23,/4/,2
   2    as capacity for up to 4 megabytes of memory, 124 gemeral-
>FI /gemeral/,1-
   2    as capacity for up to 4 megabytes of memory, 124 gemeral-
>CH /meral/,/neral/
   2    as capacity for up to 4 megabytes of memory, 124 general-
>PE
   2   >as capacity for up to 4 megabytes of memory, 128 general-
   2.01>
>O MO=HEX
>T 3
   3     707572706F7365207265676973746572732C206D756C74606C6576656C20696E746
         72757074207374727563747572652C2020202020202020202020202020202020202
>SU /6D75/6D6A
   3     707572706F7365207265676973746572732C206D6A6C74606C6576656C20696E746
         727570742073747275637475726552C2020202020202020202020202020202020202
>O MO=A
>INC 3,,5
>TY 1-6
   1    The standard processor includes such high-performance features
   2    as capacity for up to 4 megabytes of memory, 128 general-
   3    purpose registers, multilevel interrupt structure,
   3.01 8 megabytes/second direct memory access bandwidth, dynamic
   4    memory relocation and protection hardware along with a
   6    comprehensive instruction set.
>INS 2
   2.01>as capacity for up  to 4 megabytes of memory, 128 general-
   2.02>
>TY 1-3
   1    The standard processor includes such high-performance features
   2    as capacity for up to 4 megabytes of memory, 128 general-
   2.01 as capacity for up to 4 megabytes of memory, 128 general-
   3    purpose registers, multilevel interrupt structure,
>DEL 2.01
>TY 1-6
   1    The standard processor includes such high-performance features
   2    as capacity for up to 4 megabytes of memory, 128 general-
   3    purpose registers, multilevel interrupt structure,
   4    8 megabytes/second direct memory access bandwidth, dynamic
   5    memory relocation and protection hardware along with a
   6    comprehensive instruction set.
>COL 28,/o/,9-11
   9    Cache Memory          oWritable Control Store
  10    Memory Error Logger   oFloating Point Processor
  11    Battery Backup System oData Handling Instructions
```

```
>CH /o/,//,9-11
    9    Cache Memory            Writable Control Store
   10    Memory Error Logger     Floating Point Processor
   11    Battery Backup System   Data Handling Instructions
>SA *
 WORK FILE = M300:CH4EDT.000/P
 RENUMBERED INPUT FILE AVAILABLE, M300:CH4EDT.008/P
>G ch4edt.008
>TY 1-
     1   The standard processor includes such high-performance features
     2   as capacity for up to 4 megabytes of memory, 128 general-
     3   purpose registers, multilevel interrupt structure,
     4   8 megabytes/second direct memory access bandwidth, dynamic
     5   memory relocation and protection hardware along with a
     6   comprehensive instruction set.
     7   To complement the 3220's inherent power, Perkin-Elmer offers
     8   a host of performance instruction sets.
     9   Cache Memory            Writable Control Store
    10   Memory Error Logger     High Performance Floating Point
    11   Battery Backup System   High Speed Data Handling Instructions
>END
MAUREEN -END OF TASK CODE=  0   CPUTIME=7.313/6.777
```

# APPENDIX A
# COMMAND SUMMARY

$$\underline{\text{AF}}\text{TER /string/newstring}\left[\text{/}\left[\begin{Bmatrix}\text{line no}_1\\ \text{/string}_1\text{/[col]}\end{Bmatrix}\right]\left[-\left[\begin{Bmatrix}\text{line no}_n\\ \text{/string}_2\text{/[col]}\end{Bmatrix}\right]\right]\right]$$

$\underline{\text{AG}}\text{AIN}$

$\underline{\text{AP}}\text{PEND}$

$$\underline{\text{BE}}\text{FORE /string/newstring}\left[\text{/}\left[\begin{Bmatrix}\text{line no}_1\\ \text{/string}_1\text{/[col]}\end{Bmatrix}\right]\left[-\left[\begin{Bmatrix}\text{line no}_n\\ \text{/string}_2\text{/[col]}\end{Bmatrix}\right]\right]\right]$$

$\underline{\text{BO}}\text{TTOM}$

$$\underline{\text{CH}}\text{ANGE /oldstring/[col],/newstring/}$$
$$\left[,\begin{Bmatrix}\begin{Bmatrix}\text{line no}\\ \text{/string/[col]}\end{Bmatrix}\\ \left[\begin{Bmatrix}\text{line no}_1\\ \text{/string}_1\text{/[col]}\end{Bmatrix}\right]-\left[\begin{Bmatrix}\text{line no}_n\\ \text{/string}_2\text{/[col]}\end{Bmatrix}\right]\end{Bmatrix}\right]$$

$$\underline{\text{CO}}\text{LUMN col no,/string/}\left[,\begin{Bmatrix}\begin{Bmatrix}\text{line no}\\ \text{/string/[col]}\end{Bmatrix}\\ \left[\begin{Bmatrix}\text{line no}_1\\ \text{/string}_1\text{/[col]}\end{Bmatrix}\right]-\left[\begin{Bmatrix}\text{line no}_n\\ \text{/string}_2\text{/[col]}\end{Bmatrix}\right]\end{Bmatrix}\right]$$

```
         ┌(line no1          )┐ ┌   ┌(line non          )┐┐
DELETE   │{                  }│ │ - │{                  }││
         └(/string1/[col])   )┘ └   └(/string2/[col])   )┘┘

DONE

END

                         ┌ ┌(line no          )                            ┐┐
                         │ │{                 }                            ││
                         │ │(/string/[col])   )                            ││
FIND /string/[col]       │,{                                              } │
                         │ │┌(line no1        )┐  ┌(line non          )┐ ││
                         │ ││{                 }│ -│{                  }│ ││
                         └ └└(/string1/[col]) )┘  └(/string2/[col])   )┘ ┘┘

GET fd

      ┌(mnemonic)┐
HELP  │{        }│
      └(   *    )┘

          ┌(line no          )┐
INCLUDE   │{                 }│ ,[fd]
          └(/string/[col])   )┘

          ┌ ┌(line no          )                            ┐┐
          │ │{                 }                            ││
          │ │(/string/[col])   )                            ││
          │,{                                              } │
          │ │┌(line no1        )┐  ┌(line non          )┐ ││
          │ ││{                 }│ -│{                  }│ ││
          └ └└(/string1/[col]) )┘  └(/string2/[col])   )┘ ┘┘

          ┌(line no          )┐
INSERT    │{/string/ [col]    }│
          └(current line     )┘
```

```
MOVE  ┌┌line no        ┐┐
      └└/string/[col]  ┘┘

      ┌ ┌┌line no        ┐                                                  ┐┐
      │ │└/string/[col]  ┘                                                  ││
      │,│                                                                   ││
      │ │[┌line no1           ┐] - [┌line non           ┐]                  ││
      └ └[└/string1/[col]     ┘]   [└/string2/[col]    ┘]                  ┘┘

NOFIND /string/[col]

      ┌ ┌┌line no        ┐                                                  ┐┐
      │ │└/string/[col]  ┘                                                  ││
      │,│                                                                   ││
      │ │[┌line no1           ┐] - [┌line non           ┐]                  ││
      └ └[└/string1/[col]     ┘]   [└/string2/[col]    ┘]                  ┘┘

OPTION [COMMAND[=fd]]  [,BLOCK[={n }]]  [,MODE[={HEXADECIMAL}]]
                               [  {5 }]         [ {ASCII      }]

       [,LENGTH[={n }]]  [,LIST[={          fd          }]]
       [       [ {80}]]  [      [ {fd in START command  }]]

       [ {LOG[={          fd          }]} ]
       [,{     [ {fd in START command }]} ]
       [ {NOLOG                         } ]

       [ {TAB[={tab char,col1 [,col2,...,coln]}]} ]
       [,{     [{FORTRAN                       }]} ]
       [ {     [{CAL                           }]} ]
       [ {NOTAB                                  } ]

       [,TERMINATOR[={   string    }]]  [,SCREEN[={n }]]
       [           [ {     cr      }]]  [       [ {23}]]
       [           [ {unspecified  }]]

       [ {NOVERIFY} ]
       [,{VERIFY  } ]

PAUSE
```

$$\underline{\text{RE}}\text{PLACE}\ \left[\begin{Bmatrix}\text{line no}_1\\ \text{/string}_1\text{/}[\text{col}]\end{Bmatrix}\right]\ \left[-\left[\begin{Bmatrix}\text{line no}_n\\ \text{/string}_2\text{/}[\text{col}]\end{Bmatrix}\right]\right]$$

$$\underline{\text{RU}}\text{LER}$$

$$\underline{\text{SA}}\text{VE}\ \begin{Bmatrix}\text{fd}\\ *\end{Bmatrix}\left[,\begin{Bmatrix}\begin{Bmatrix}\text{line no}\\ \text{/string/}[\text{col}]\end{Bmatrix}\\ \left[\begin{Bmatrix}\text{line no}_1\\ \text{/string}_1\text{/}[\text{col}]\end{Bmatrix}\right]-\left[\begin{Bmatrix}\text{line no}_n\\ \text{/string}_2\text{/}[\text{col}]\end{Bmatrix}\right]\end{Bmatrix}\right]$$

$$\underline{\text{SC}}\text{REEN}\ \left[\begin{Bmatrix}\text{E}\\ \text{C}\\ \text{S}\end{Bmatrix}\right]$$

$$\underline{\text{SEN}}\text{D}\ \underline{\text{STOP}}$$

$$\underline{\text{SU}}\text{BSTITUTE /oldstring/newstring}\left[\text{/}\left[\begin{Bmatrix}\text{line no}_1\\ \text{/string}_1\text{/}[\text{col}]\end{Bmatrix}\right]\left[-\left[\begin{Bmatrix}\text{line no}_n\\ \text{/string}_2\text{/}[\text{col}]\end{Bmatrix}\right]\right]\right]$$

$$\underline{\text{TO}}\text{P}$$

$$\underline{\text{TR}}\text{UNCATE}\ [\text{col no}]$$

$$\underline{\text{TY}}\text{PE}\ \left[\begin{Bmatrix}\text{line no}_1\\ \text{/string}_1\text{/}[\text{col}]\end{Bmatrix}\right]\ \left[-\left[\begin{Bmatrix}\text{line no}_n\\ \text{/string}_2\text{/}[\text{col}]\end{Bmatrix}\right]\right]$$

$$+\ [\text{n}]$$

$$-\ [\text{n}]$$

$$\text{n}\ [\text{data}]$$

# APPENDIX B
# MESSAGES

!!FILE ERR,LUN,SIZE ERROR,fd

The specified block size exceeds the system maximum data block size.

!!I/O ERR, LUn, code, fd

| | |
|---|---|
| n | is the logical unit (lu) number. |
| code | is the error code. See Table B-1. |
| fd | is the file descriptor. |

TABLE B-1 SVC1 ERROR CODES

| ERROR CODE | MEANING |
|---|---|
| X'81' | ILLEGAL OR UNASSIGNED LU |
| X'82' | PARITY OR RECOVERABLE ERROR |
| X'84' | UNRECOVERABLE ERROR |
| X'88' | END OF FILE |
| X'90' | END OF MEDIUM |
| X'A0' | DEVICE UNAVAILABLE |
| X'C0' | ILLEGAL FUNCTION |

!!FILE ERR, LUn, code, fd

| | |
|---|---|
| n | is the lu number. |
| code | is the error code. See Table B-2. |
| fd | is the file descriptor. |

TABLE B-2 SVC7 ERROR CODES

| ERROR CODE | MEANING |
|---|---|
| 01 | ILLEGAL FUNCTION |
| 02 | LU ERROR |
| 03 | VOLUME ERROR |
| 04 | NAME ERROR |
| 05 | SIZE ERROR |
| 06 | PROTECT ERROR |
| 07 | PRIVILEGE ERROR |
| 08 | BUFFER ERROR |
| 09 | ASSIGNMENT ERROR |
| 0A | TYPE ERROR |
| 0B | FILE DESCRIPTOR ERROR |
| 0C | TGD ASSIGNMENT ERROR |
| 0D | ACCOUNT ERROR |

ALL LINES DELETED

All lines were deleted from a file using the DELETE command.

?COL # WITHIN STRING OPERAND INVALID HERE

A column was specified where it is not allowed in the command format.

?COLUMN INCORRECT

A column was specified where it is not allowed in the command format.

?COMMA MISSING

A comma was missing in the command syntax.

?COMMAND NOT RECOGNIZED

The command mnemonic was entered incorrectly.

!COMMAND LEVELS NESTED TOO DEEP

Command level exceeds that supported by program.

DELETE AND REALLOCATE VOLN:FILENAME /P?

The file specified in the SAVE command already exists or is the name of a current input file. Enter YES if the file is to be deleted and reallocated. Enter NO if the file is not to be deleted and reallocated.

!END OF COMMAND FILE

An EOF condition occurred while reading a command file.

END OF TASK CODE=0

Normal completion of a job.

END OF TASK CODE=1

Abnormal job termination as a result of invalid START arguments.

END OF TASK CODE=2

Execution was terminated as a result of an error encountered in batch processing.

?FILE NOT FOUND

The file was not found on the specified volume or the default volume.

FILE NOT SAVED

NO was entered in response to the DELETE AND REALLOCATE message.

?FILENAME INCORRECT

The filename was entered incorrectly.

!FIRST LINE

The line number specified exceeds top-of-file.

HELP FILE FORMAT ERROR - ABORTED

The Help file format is incorrect.

HELP FILE NOT FOUND

The Help file was not found in the system volume under the system account.

!INVALID BLOCK SIZE

The specified block size is not in the range allowed.

?INVALID COLUMN SPECIFIER

The specified column number is invalid.

?INVALID HEX CHARACTER

The specified character string contains an invalid hex character.

!INVALID LIST DEVICE

The fd specified in the list command cannot be assigned. The original assignment is in effect.

!INVALID SEND MESSAGE

An invalid parameter was entered during a send sequence.

!INVALID TRUNCATE VALUE

An invalid value was specified for the truncate column.

!INVALID OPERAND

The parameter specified in the SCREEN command is not one of 'S', 'E', or 'C'.

!FILE RECORD LENGTH=m CURRENT LINE LENGTH=n

The record length of the file specified when a GET command was issued, exceeds the length specified in the OPTION LENGTH command.

!LAST LINE

The number specified results in going to end-of-file.

!LENGTH OPTION NOT ACCEPTED

The length option was specified after the GET command.

!LINE NOT FOUND

The line or string after which data is to be inserted could not be found.

?LINE OPERAND INVALID

The specified line number is invalid.

!NON-BLANK CHARACTERS LOST

The replacement character string exceeds the default number of characters per line. Non-blank characters are truncated.

!NO TEXT

An Edit command was entered before a file was brought off disc via the GET command, or there was no data in the allocated file.

?NUMBER INVALID

The line parameter is not a numeric character.

?OCCURS IN ALL LINES

When executing the NOFIND command, the specified character string to be searched for was found in all lines.

!OPTION ERROR FOLLOWS XXXX

The characters appearing in place of xxxx are the last four non-blank characters recognized as valid. An OPTION parameter was not valid.

!RANGE 1 NOT FOUND

The lower limit of the range could not be found.

!RANGE 2 NOT FOUND

The upper limit of the range could not be found.

?RANGE INVALID

The specified range was entered incorrectly.

!REMINDER - SAVE YOUR CURRENT TEXT

An END or another GET command was entered before entering a SAVE of current data.

?REPEAT DISALLOWED

The DELETE, END, GET commands, and a line of data entered with a line number are not re-executed by entering a carriage return.

!SCREEN SIZE INVALID

The number of lines specified for screen size is invalid.

?STRING INVALID

The specified character string was entered incorrectly.

!STRING NOT FOUND

The character string was not found within the specified range.

!TAB ILLEGAL

An illegal tab setting was specified, or an illegal tab character was used.

!TABS TABLE FULL

There are more than 20 tab settings.

!TRUNCATE VALUE INVALID

There is an invalid value specified for the truncate column.

?UNABLE TO LIST FULL SCREEN

There is insufficient data in the file to list a full screen as requested in the SCREEN command.

# APPENDIX C
# ASCII - EBCDIC CONVERSION TABLES

| CHARACTER | CARD PUNCHES | ASCII | EBCDIC |
|---|---|---|---|
| | | HEXADECIMAL | |
| **Uppercase Letters** | | | |
| A | 12-1 | 41 | C1 |
| B | 12-2 | 42 | C2 |
| C | 12-3 | 43 | C3 |
| D | 12-4 | 44 | C4 |
| E | 12-5 | 45 | C5 |
| F | 12-6 | 46 | C6 |
| G | 12-7 | 47 | C7 |
| H | 12-8 | 48 | C8 |
| I | 12-9 | 49 | C9 |
| J | 11-1 | 4A | D1 |
| K | 11-2 | 4B | D2 |
| L | 11-3 | 4C | D3 |
| M | 11-4 | 4D | D4 |
| N | 11-5 | 4E | D5 |
| O | 11-6 | 4F | D6 |
| P | 11-7 | 50 | D7 |
| Q | 11-8 | 51 | D8 |
| R | 11-9 | 52 | D9 |
| S | 0-2 | 53 | E2 |
| T | 0-3 | 54 | E3 |
| U | 0-4 | 55 | E4 |
| V | 0-5 | 56 | E5 |
| W | 0-6 | 57 | E6 |
| X | 0-7 | 58 | E7 |
| Y | 0-8 | 59 | E8 |
| Z | 0-9 | 5A | E9 |
| **Lowercase Letters** | | | |
| a | 12-0-1 | 61 | 81 |
| b | 12-0-2 | 62 | 82 |
| c | 12-0-3 | 63 | 83 |
| d | 12-0-4 | 64 | 84 |
| e | 12-0-5 | 65 | 85 |
| f | 12-0-6 | 66 | 86 |
| g | 12-0-7 | 67 | 87 |
| h | 12-0-8 | 68 | 88 |
| i | 12-0-9 | 69 | 89 |
| j | 12-11-1 | 6A | 91 |
| k | 12-11-2 | 6B | 92 |
| l | 12-11-3 | 6C | 93 |

| CHARACTER | CARD PUNCHES | ASCII | EBCDIC |
|---|---|---|---|
| | | HEXADECIMAL | |
| **Lowercase Letters** | | | |
| m | 12-11-4 | 6D | 94 |
| n | 12-11-5 | 6E | 95 |
| o | 12-11-6 | 6F | 96 |
| p | 12-11-7 | 70 | 97 |
| q | 12-11-8 | 71 | 98 |
| r | 12-11-9 | 72 | 99 |
| s | 11-0-2 | 73 | A2 |
| t | 11-0-3 | 74 | A3 |
| u | 11-0-4 | 75 | A4 |
| v | 11-0-5 | 76 | A5 |
| w | 11-0-6 | 77 | A6 |
| x | 11-0-7 | 78 | A7 |
| y | 11-0-8 | 79 | A8 |
| z | 11-0-9 | 7A | A9 |
| **Numerals** | | | |
| 0 | 0 | 30 | F0 |
| 1 | 1 | 31 | F1 |
| 2 | 2 | 32 | F2 |
| 3 | 3 | 33 | F3 |
| 4 | 4 | 34 | F4 |
| 5 | 5 | 35 | F5 |
| 6 | 6 | 36 | F6 |
| 7 | 7 | 37 | F7 |
| 8 | 8 | 38 | F8 |
| 9 | 9 | 39 | F9 |
| **Symbols** | | | |
| Exclamation point ! | 11-8-2 | 5D | 5A |
| Quotation mark, dieresis " | 8-7 | 22 | 7F |
| Number sign, pound sign # | 8-3 | 23 | 7B |
| Dollar sign $ | 11-8-3 | 24 | 5B |
| Percent sign % | 0-8-4 | 25 | 6C |
| Ampersand & | 12 | 26 | 50 |
| Apostrophe, acute accent ' | 8-5 | 27 | 7D |
| Opening parenthesis ( | 12-8-5 | 28 | 4D |
| Closing parenthesis ) | 11-8-5 | 29 | 5D |
| Asterisk * | 11-8-4 | 2A | 5C |
| Plus sign + | 12-8-6 | 2B | 4E |
| Comma, cedilla , | 0-8-3 | 2C | 6B |
| Minus sign, hyphen - | 11 | 2D | 60 |
| Period, decimal point . | 12-8-3 | 2E | 4B |
| Slash, virgule, solidus / | 0-1 | 2F | 61 |
| Colon : | 8-2 | 3A | 7A |
| Semicolon ; | 11-8-6 | 3B | 5E |

| CHARACTER | | CARD PUNCHES | ASCII | EBCDIC |
|---|---|---|---|---|
| | | | HEXADECIMAL | |
| **Symbols** | | | | |
| Less than | < | 12-8-4 | 3C | 4C |
| Equal sign | = | 8-6 | 3D | 7E |
| Greater than | > | 0-8-6 | 3E | 6E |
| Question mark | ? | 0-8-7 | 3F | 6F |
| Commercial at symbol | @ | 8-4 | 40 | 7C |
| Opening bracket | [ | 12-8-2 | 5B | 4A |
| Closing bracket | ] | 12-8-7 | 21 | 4F |
| Reverse slash | \ | 0-8-2 | 5C | E0 |
| Circumflex | ^ | 11-8-7 | 5E | 5F |
| Underline | _ | 0-8-5 | 5F | 6D |
| Grave accent | ` | 8-1 | 60 | 79 |
| Opening brace | { | 12-0 | 7B | C0 |
| Closing brace | } | 11-0 | 7D | D0 |
| Vertical line | \| | 12-11 | 7C | 6A |
| Overline, tilde | ‾ | 11-0-1 | 7E | A1 |
| **Nonprintable Characters** | | | | |
| ACK (acknowledge) | | 0-9-8-6 | 06 | 2E |
| BEL (bell) | | 0-9-8-7 | 07 | 2F |
| BS (backspace) | | 11-9-6 | 08 | 16 |
| CAN (cancel) | | 11-9-8 | 18 | 18 |
| CR (carriage return) | | 12-9-8-5 | 0D | 0D |
| DC1 (device control 1) | | 11-9-1 | 11 | 11 |
| DC2 (device control 2) | | 11-9-2 | 12 | 12 |
| DC3 (device control 3) | | 11-9-3 | 13 | 13 |
| DC4 (device control 4) | | 9-8-4 | 14 | 3C |
| DEL (delete) | | 12-9-7 | 7F | 07 |
| DLE (data link escape) | | 12-11-9-8-1 | 10 | 10 |
| EM (end of medium) | | 11-9-8-1 | 19 | 19 |
| ENQ (enquiry) | | 0-9-8-5 | 05 | 2D |
| ECT (end of transmission) | | 9-7 | 04 | 37 |
| ESC (escape) | | 0-9-7 | 1B | 27 |
| ETB (end of transmission block) | | 0-9-6 | 17 | 26 |
| ETX (end of text) | | 12-9-3 | 03 | 03 |
| FF (form feed) | | 12-9-8-4 | 0C | 0C |
| FS (file separator) | | 11-9-8-4 | 1C | 1C |
| GS (group separator) | | 11-9-8-5 | 1D | 1D |
| HT (horizontal tabulation) | | 12-9-5 | 09 | 05 |
| LF (line feed) | | 0-9-5 | 0A | 25 |
| NAK (negative acknowledge) | | 9-8-5 | 15 | 3D |
| NUL (null) | | 12-0-9-8-1 | 00 | 00 |
| RS (record separator) | | 11-9-8-6 | 1E | 1E |
| SI (shift in) | | 12-9-8-7 | 0F | 0F |
| SC (shift out) | | 12-9-8-6 | 0E | 0E |
| SOH (start of heading) | | 12-9-1 | 01 | 01 |
| SP (space) | | | 20 | 40 |
| STX (start of text) | | 12-9-2 | 02 | 02 |

| CHARACTER | CARD PUNCHES | ASCII | EBCDIC |
|---|---|---|---|
| | | HEXADECIMAL | |
| Nonprintable Characters | | | |
| SUB (substitute) | 9-8-7 | 1A | 3F |
| SYN (synchronous idle) | 9-2 | 16 | 32 |
| US (unit separator) | 11-9-8-7 | 1F | 1F |
| VT (vertical tabulation) | 12-9-8-3 | 0B | 0B |

# INDEX

# INDEX