

EDT Editor Manual

Order No. AA-J726A-TC

October 1980

This manual describes how to use the EDT interactive text editor and serves as a reference source. The manual is intended for all users of EDT.

SOFTWARE VERSION: EDT V2.0

digital equipment corporation, maynard, massachusetts

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license, and may only be used or copied in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by DIGITAL or its affiliated companies.

Copyright © 1980 Digital Equipment Corporation

The postage-paid READER'S COMMENTS form on the last page of this document requests your critical evaluation to assist us in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

| | |
|--------------|--------|
| DEC | FOCAL |
| DECnet | IAS |
| DECsystem-10 | PDP |
| DECtape | RSTS |
| DECUS | RSX |
| DIBOL | UNIBUS |
| DIGITAL | VAX |
| | VMS |

Contents

| | Page |
|--|------|
| Preface | ix |
| Summary of New Features and Technical Changes | xiii |
| Chapter 1 Introduction to EDT | |
| Overview | 1-1 |
| How to Use This Manual | 1-1 |
| Your Editing Session | 1-2 |
| EDT Editing Features | 1-3 |
| HELP Facility | 1-3 |
| File Protection by Journaling | 1-4 |
| Startup Command Files | 1-5 |
| Keypad, Line, and Nokeypad Editing | 1-6 |
| Line Numbers | 1-7 |
| Multiple Files and Buffers | 1-8 |
| Redefining Keys | 1-9 |
| Types of EDT Commands | 1-9 |
| SET and SHOW Commands | 1-9 |
| Keypad Editing Commands | 1-9 |
| Line Editing Commands | 1-9 |
| Nokeypad Editing Commands | 1-10 |
| Chapter 2 Sample Editing Sessions | |
| Getting Started | 2-1 |
| Editing Text in Buffers | 2-1 |
| The Command Line | 2-2 |
| Starting an EDT Session | 2-3 |
| Keypad Editing | 2-4 |
| Starting Keypad Editing | 2-5 |
| The HELP Facility | 2-5 |
| Keypad Functions | 2-8 |
| Inserting Text | 2-8 |
| Moving the Cursor | 2-9 |
| Using Select Ranges to Move Text | 2-12 |
| Deleting Words and Characters | 2-14 |
| Deleting Lines | 2-15 |
| How to Change Case | 2-18 |
| Ending a Keypad Editing Session | 2-19 |
| Line Editing | 2-20 |
| Creating Files | 2-20 |
| Using Line Editing Commands | 2-25 |
| Revising Files | 2-28 |
| Multiple Files and Buffers | 2-32 |
| Nokeypad Editing | 2-36 |
| Starting Nokeypad Editing | 2-36 |
| Nokeypad Editing Commands | 2-38 |

Chapter 3 Protecting Your Editing Session

| | |
|--|-----|
| The Journal File and RECOVER Qualifier | 3-1 |
| Editing the Journal File | 3-4 |
| Error Messages | 3-7 |

Chapter 4 The Command Line and Startup Command Files

| | |
|---|-----|
| The DCL and PDS Command Line | 4-2 |
| DCL and PDS Command Line Qualifiers | 4-2 |
| The MCR and CCL Command Line | 4-4 |
| MCR and CCL Command Line Qualifiers | 4-5 |
| Using Command Lines and Qualifiers | 4-7 |
| Startup Command Files | 4-9 |

Chapter 5 Keypad Editing

| | |
|--|------|
| Starting an Editing Session | 5-1 |
| Using the Keypad | 5-2 |
| Essential Functions | 5-7 |
| GOLD | 5-7 |
| HELP | 5-7 |
| RESET | 5-8 |
| Refreshing the Screen (CTRL/W) | 5-8 |
| Returning to Line Editing (CTRL/Z) | 5-8 |
| Ending an Editing Session | 5-9 |
| Saving Your Edits | 5-9 |
| Deleting Your Edits | 5-9 |
| Inserting Text | 5-10 |
| Inserting Characters | 5-10 |
| Using OPEN LINE | 5-11 |
| Moving the Cursor | 5-12 |
| The Arrow Keys | 5-13 |
| UP | 5-13 |
| DOWN | 5-13 |
| LEFT | 5-14 |
| RIGHT | 5-14 |
| Setting the Direction of Cursor Movement | 5-15 |
| ADVANCE | 5-15 |
| BACKUP | 5-15 |
| Movement by Entity | 5-16 |
| CHAR (Character) | 5-16 |
| WORD | 5-17 |
| EOL (End Of Line) | 5-18 |
| LINE | 5-18 |
| BACK SPACE | 5-19 |
| Movement Throughout the Buffer | 5-20 |
| PAGE | 5-20 |
| SECTION | 5-20 |
| TOP | 5-21 |
| BOTTOM | 5-21 |

| | |
|--|------|
| Locating Text | 5-21 |
| FIND | 5-21 |
| FNDNXT (Find Next) | 5-23 |
| Deleting and Reinserting Text | 5-25 |
| Deleting and Reinserting by Character | 5-25 |
| DELETE | 5-25 |
| DEL C (Delete Character) | 5-26 |
| UND C (Undelete Character) | 5-27 |
| Deleting and Reinserting by Word | 5-28 |
| LINE FEED | 5-28 |
| DEL W (Delete Word) | 5-28 |
| UND W (Undelete Word) | 5-29 |
| Deleting and Reinserting by Line | 5-30 |
| CTRL/U | 5-30 |
| DEL EOL (Delete to End of Line) | 5-30 |
| DEL L (Delete through End of Line) | 5-32 |
| UND L (Undelete Line) | 5-32 |
| Selecting and Moving Text | 5-32 |
| SELECT | 5-33 |
| CUT | 5-33 |
| PASTE | 5-33 |
| Using SELECT, CUT, and PASTE | 5-33 |
| APPEND | 5-38 |
| Replacing and Substituting Text | 5-40 |
| REPLACE | 5-40 |
| SUBS | 5-44 |
| Using Line Editing Commands | 5-46 |
| ENTER | 5-46 |
| COMMAND | 5-47 |
| Using COMMAND and ENTER | 5-47 |
| Correcting Mistakes When You Enter Commands | 5-47 |
| Special Characters, Changing Case, and Filling Lines | 5-48 |
| SPECINS | 5-48 |
| CHNGCASE | 5-48 |
| FILL | 5-49 |
| Setting Tabs | 5-53 |
| CTRL/A | 5-53 |
| CTRL/E | 5-53 |
| CTRL/D | 5-53 |
| CTRL/T | 5-53 |
| CTRL/K and CTRL/C | 5-54 |
| CTRL/K | 5-54 |
| CTRL/C | 5-54 |

Chapter 6 Line Numbers, Text Buffers, and Ranges

| | |
|---|-----|
| Line Numbers | 6-1 |
| Numbers Displayed in Line Editing | 6-1 |

| | |
|--|------|
| Resequencing Lines | 6-3 |
| Fixed Line Numbers | 6-3 |
| Text Buffers | 6-5 |
| Main Buffer | 6-6 |
| Paste Buffer | 6-6 |
| Additional Text Buffers | 6-6 |
| Range Specifications | 6-7 |
| Single Line Ranges | 6-7 |
| Contiguous Line Ranges | 6-10 |
| Noncontiguous Ranges | 6-11 |
| Text Buffer Range Specifications | 6-11 |

Chapter 7 Line Editing

| | |
|---------------------------|------|
| CHANGE | 7-1 |
| CLEAR | 7-2 |
| COPY | 7-3 |
| DEFINE KEY | 7-4 |
| DEFINE MACRO | 7-4 |
| DELETE | 7-6 |
| EXIT | 7-8 |
| FIND | 7-8 |
| HELP | 7-9 |
| INCLUDE | 7-10 |
| INSERT | 7-11 |
| MOVE | 7-11 |
| Null (Implied TYPE) | 7-12 |
| PRINT | 7-13 |
| QUIT | 7-14 |
| REPLACE | 7-14 |
| RESEQUENCE | 7-15 |
| SUBSTITUTE | 7-16 |
| SUBSTITUTE NEXT | 7-19 |
| TYPE | 7-20 |
| WRITE | 7-22 |

Chapter 8 Nokeypad Editing

| | |
|----------------------------------|------|
| Entities of Text | 8-2 |
| Nokeypad Command Structure | 8-4 |
| Nokeypad Commands | 8-5 |
| ADV (Advance) | 8-5 |
| APPEND | 8-5 |
| ASC (ASCII) | 8-5 |
| BACK (Backup) | 8-6 |
| CHGC (Change Case) | 8-6 |
| CUT | 8-7 |
| D (Delete) | 8-9 |
| DEFK (Define Key) | 8-10 |
| EX (Exit) | 8-10 |
| EXT (Extended) | 8-10 |
| FILL | 8-11 |
| I (Insert) | 8-12 |
| Null (Implied TYPE) | 8-13 |
| PASTE | 8-15 |

| | |
|---------------------------------|------|
| QUIT | 8-15 |
| R (Replace) | 8-15 |
| REF (Refresh) | 8-16 |
| S/s1/s2/ (Substitute) | 8-16 |
| SEL (Select) | 8-17 |
| SHL (Shift Left) | 8-18 |
| SHR (Shift Right) | 8-18 |
| SN (Substitute Next) | 8-19 |
| TAB | 8-21 |
| TADJ (Tab Adjust) | 8-22 |
| TC (Tab Compute) | 8-23 |
| TD (Tab Decrement) | 8-24 |
| TI (Tab Increment) | 8-24 |
| TOP | 8-24 |
| UNDC (Undelete Character) | 8-24 |
| UNDW (Undelete Word) | 8-24 |
| UNDL (Undelete Line) | 8-25 |
| ^ (Circumflex) | 8-25 |

Chapter 9 The Set and Show Commands

| | |
|-----------------------------|------|
| The SET Commands | 9-2 |
| SET CASE | 9-2 |
| SET CURSOR top:bottom | 9-3 |
| SET ENTITY | 9-4 |
| SET KEYPAD | 9-5 |
| SET LINES number | 9-5 |
| SET MODE | 9-5 |
| SET MODE CHANGE | 9-5 |
| SET MODE LINE | 9-5 |
| SET [NO]NUMBERS | 9-6 |
| SET [NO]QUIET | 9-6 |
| SET SCREEN width | 9-6 |
| SET SEARCH | 9-7 |
| SET SEARCH EXACT | 9-7 |
| SET SEARCH BOUNDED | 9-7 |
| SET SEARCH END | 9-7 |
| SET TAB n | 9-8 |
| SET TERMINAL | 9-12 |
| SET [NO]TRUNCATE | 9-12 |
| SET [NO]VERIFY | 9-13 |
| SET [NO]WRAP n | 9-14 |
| The SHOW Commands | 9-15 |
| SHOW BUFFER | 9-15 |
| SHOW CASE | 9-15 |
| SHOW CURSOR | 9-16 |
| SHOW ENTITY | 9-16 |
| SHOW KEY | 9-16 |
| SHOW SCREEN | 9-17 |
| SHOW SEARCH | 9-17 |
| SHOW TERMINAL | 9-17 |
| SHOW VERSION | 9-17 |

Chapter 10 Redefining Keys

| | |
|----------------------------|------|
| Key Concepts | 10-1 |
| DEFINE KEY Command | 10-2 |
| CTRL/K | 10-5 |
| Using a Repeat Count | 10-9 |

Appendix A Error Messages

Appendix B ASCII Decimal Equivalents

Appendix C Terminals with Low Baud Rates

Glossary

Index

Figures

| | |
|---|------|
| 1-1: Sample HELP Description | 1-3 |
| 1-2: Keypad Editing Help Text (VT100) | 1-4 |
| 1-3: VT100 Keyboard and Keypad | 1-6 |
| 2-1: VT100 Keypad Functions | 2-6 |
| 2-2: VT52 Keypad Functions | 2-7 |
| 5-1: The VT52 Keypad and Its Functions | 5-3 |
| 5-2: The VT100 Keypad and Its Functions | 5-4 |
| 10-3: Keypad Numbers | 10-3 |

Tables

| | |
|--|------|
| 2-1: Operating System CLIs and Prompts | 2-3 |
| 4-1: DCL and PDS Command Line Qualifiers | 4-3 |
| 4-2: MCR and CCL Command Line Qualifiers | 4-6 |
| 6-1: Single Line Ranges | 6-8 |
| 6-2: Contiguous Line Ranges | 6-10 |
| 8-1: Entities of Text | 8-2 |
| 9-1: SET and SHOW Parameters | 9-1 |

Preface

Audience

The *EDT Editor Manual* is intended for anyone who wants to use the EDT editor. The manual shows how to create and update files with the editor and serves as a reference source.

You do not need experience with computers to use this manual. However, you should have:

- Access to an operating system with EDT
- The ability to log in and out

If you do not understand some of the terms used in this manual, check the Glossary at the end of the manual for definitions. Consult the manuals in your system's documentation set to learn about file handling outside of EDT.

Document Structure

The "Summary of New Features and Technical Changes" describes how EDT has been improved since Version 1.0.

Chapters 1 and 2 introduce the editor and guide you through sample sessions of editing with EDT. Chapter 3 describes how to use the journaling feature to protect the contents of your files while you edit them. Chapter 4 describes the command line and command files that you can use when you start an editing session.

Chapter 5 describes keypad editing and shows examples of each editing function. Chapter 6 explains line numbers, text buffers, and ranges, which you use for line editing (Chapter 7) and nokeypad editing (Chapter 8).

Chapter 9 describes the SET and SHOW commands, which let you specify and check editing characteristics, such as the width of a display of text across the screen.

Chapter 10 describes redefining keys, which is an advanced feature that lets you assign different functions to keypad keys.

Appendixes A, B, and C describe error messages, ASCII character codes, and information on terminals with low baud rates, respectively.

Conventions


This manual uses the following symbols:



Symbol

Meaning



A rectangle represents the cursor in examples.

CTRL/x or 

The phrase CTRL/x shows that you must press the key labeled CTRL while you simultaneously press another key; for example,  or .



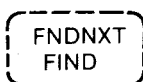
The DELETE key symbol shows that you must press the DELETE key to erase the character to the left of the cursor.



The SPACE key symbol shows that you must press the space bar to insert a space.



+






Dashed lines around command names indicate keypad functions keys. Two keys separated by a plus sign mean that you should press one key and then another. In this example you press the GOLD function key and then the FIND/FNDNXT key.

Pressing the GOLD key first means that the alternate of the two functions on a key will take effect (FIND in this example). Otherwise, the standard function (FNDNXT here) occurs.




Solid lines around one or more characters show an actual key on the keyboard.

GOLD/x or 

The phrase GOLD/x indicates that you must press the GOLD keypad key and then a keyboard key; for example,  or .



The RETURN key symbol shows that you must press the RETURN key, which produces a carriage return/line feed combination. Unless otherwise indicated, end all lines with a .



Brackets enclose optional parameters within command statements. Parameters describe the entity that a command affects, such as a file, line, or word. For example, [range] shows that you can enter a range of lines.

{ } Braces enclose a list of alternatives within a command, one of which you must specify. For example, {HCPY | VT52 | VT100} shows that you must specify a hardcopy, VT52, or VT100 terminal.

The OR symbol separates qualifiers for the same operation or the alternatives contained within braces. For example, 'string' | "string" means that you could type either 'string' or "string".

☞ represents your operating system's command level prompt. The actual prompting symbol (for example, DCL>, Ready, >, or \$) appears at the left of your screen.

COPY The minimum abbreviation for a command (for example, CO for COPY) is underlined in the command format. EDT accepts any input from the minimum abbreviation up through the complete command. The examples in this manual use the complete command name.

Color Red print in examples shows what you type. Red is also used on key symbols to show which function on a keypad key you use.

... A horizontal ellipsis shows indefinite repetition in a command. For example, [range AND range AND. . .] shows that you can specify several ranges.

: A vertical ellipsis shows that not all of the statements in an example or figure are shown.

Some terminology can vary from system to system. The following are examples of similar terms for different operating systems:

| VAX/VMS | PDP-11 |
|---------------------------|------------------------|
| file type | file extension |
| command line qualifier | command line switch |

Summary of New Features and Technical Changes

EDT Version 2.0 is a compatible upgrade of EDT Version 1.0. The following is a list of EDT's new features and technical changes:

- Line-oriented editing compatible with EDT Version 1.0
- Improved handling of line numbering
 - Fractional line numbers
 - Original line numbers saved
- Multiple commands per line
- Enhanced capabilities in keypad and nokeypad editing
- Journaling facility to protect your edits from system crashes
- Shareability by multiple users on time-sharing systems
- Extensive HELP facility both for each command and for each key of the keypad
- Capability of defining keys and macros

The following Version 1.0 keywords are still valid in Version 2.0. However, the percent signs (%) are necessary only when these commands are used with the Null command.

| | |
|---------|--------|
| %ALL | %FOR |
| %AND | %LAST |
| %BEFORE | %REST |
| %BEGIN | %TO |
| %BUFFER | %THRU |
| %END | %WHOLE |

In line editing, the semicolon (;) that was used in Version 1.0 to display consecutive lines has been replaced by the pound sign (#) in Version 2.0.

The following example shows how to display 100 lines, starting with line 10.

Version 1.0

*Type 10;100

Version 2.0

*Type 10#100

The semicolon in Version 2.0 is used as a statement separator or delimiter. The following command displays lines 20, 40, and 60.

Version 2.0

*Type 20;Type 40;Type 60

Chapter 1

Introduction to EDT

Overview

EDT is a *text editor* that is available on many of DIGITAL's operating systems. You can use EDT to create a file, to enter and manipulate text in the file, and to save or delete the work you do in an editing session. EDT works with text files that contain anything from programs to personal letters.

EDT offers many features to make text editing easier and more efficient. These features include:

- An on-line HELP facility that you can refer to at any time in your editing session
- Protection of your files with *journaling*
- *Startup command files*, which let you specify editing characteristics before you start EDT
- A choice of *keypad*, *line*, or *nokeypad* editing, depending on the kind of terminal you have and your preference for editing text
- Two kinds of line numbering
- The capability to work with several files or parts of files at a time
- The ability to redefine the functions of keys, which allows you to tailor EDT to meet your needs

How to Use This Manual

The kind of editing you use will determine the best order for you to read this manual.

Most people who have VT52 or VT100 terminals will want to use keypad editing. (A keypad is the set of keys at the right of the keyboard.) Keypad editing lets you perform most editing functions by pressing keys instead of typing in commands. To learn about keypad editing, read Chapters 1 through 5.

If you have a hardcopy terminal or if you prefer to edit by numbered lines, you will probably want to use line editing. In line editing, you enter commands after an asterisk prompt (*). Chapters 1 through 4 and Chapter 7 are most relevant to line editing.

Users who have video terminals without keypads may prefer nokeypad editing. Nokeypad editing lets you display whole sections of a file's contents on the video screen while you edit the file. To use nokeypad editing, read Chapters 1 through 4 and Chapter 8 before referring to the rest of the manual.

Chapters 9 and 10 describe some advanced features. When you become more familiar with EDT, you may want to read Chapter 9 for the SET and SHOW commands. These commands let you determine the characteristics of your editing session, such as the width of a display of text across the screen. Chapter 10 describes how to redefine the functions of keys to meet your needs.

If some of the terminology in this manual is unfamiliar to you, check the Glossary at the end of the manual for definitions.

Your Editing Session

Imagine that the computer creates a workspace for you when you give the command to start EDT. This workspace is known as a *buffer*, and EDT names it MAIN. You can either insert text in the buffer for a new file or copy an existing file into the buffer.

EDT also provides other types of buffers. For example, the text that you delete with the CUT command is stored in the buffer named PASTE until you replace this text with another deletion. During an editing session you can create extra buffers to store sections of text.

The original copy of any file that you edit remains unchanged. The buffer copy does not become a file until you end an editing session with the EXIT command.

At the end of an editing session, you can either save or delete the text in the buffer. If you save the buffer copy of a file you have worked on, this version becomes a new file. If you delete your edits to a file, a new file is not created and the original file remains unchanged. If you enter text for a new file in the buffer but do not save your edits, no file is created.

EDT Editing Features

The following sections describe the features of EDT.

HELP Facility

There are two ways to get help messages in EDT:

1. Using the HELP command
2. Pressing the HELP key during keypad editing

You can use the HELP command after the line editing prompt (*) or after the keypad editing COMMAND: prompt to get a detailed HELP message. (Line editing is described in Chapter 7.) To get help, type:

* HELP^(RET)

A message appears on your screen to describe:

- The format to use when requesting HELP on a specific topic
- The list of topics on which help messages are available

Figure 1-1 is an example of what you would see if you requested help on the TYPE command:

Figure 1-1: Sample HELP Description

* HELP TYPE^(RET)

TYPE

The TYPE (abbreviation: T) command displays the specified range of lines on the terminal.

Format: TYPE [range] [/BRIEF[:n]] [/STAY]

The first line in the specified range becomes the current line.

If the range specification starts with a non-alphabetic character, the keyword TYPE may be omitted completely.

Additional information available:

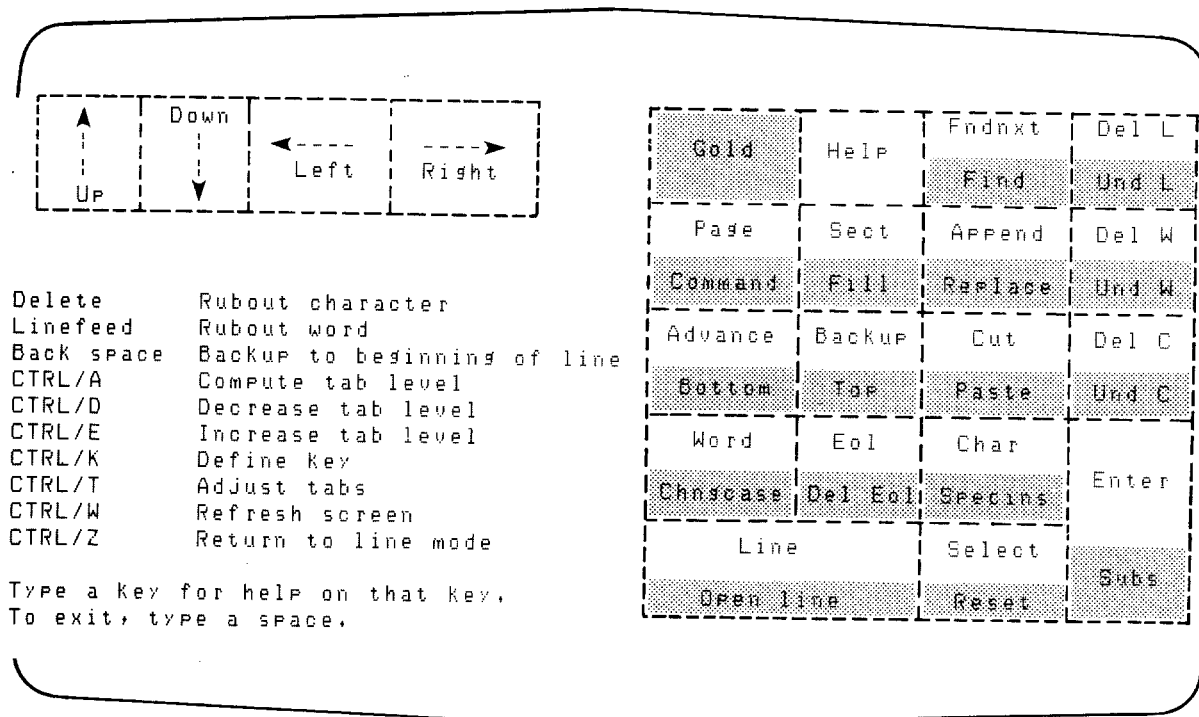
/BRIEF /STAY

*^(RET)

If you wanted help on one of the subtopics in Figure 1-1 — for example, /BRIEF — you would then enter HELP TYPE/BRIEF after the asterisk.

You can also press the HELP function key in keypad editing to see a diagram of all keypad editing functions. (Keypad editing is described in Chapter 5). Figure 1-2 shows the message that appears on the screen when you press the HELP key on a VT100 during keypad editing; a slightly different message appears on VT52s.

Figure 1-2: Keypad Editing Help Text (VT100)



At this point you can press individual keypad keys to read what each key does. When you use the HELP facility, you do not affect your editing session. Exit HELP by pressing the space bar.

File Protection by Journaling

The edits you make to the text in a buffer are protected by the *journaling* feature. If for any reason your editing session ends unexpectedly (due to system problems, for example), you can use a *journal file* to restore your work.

Unless you specify otherwise, EDT saves a copy of your work in a journal file while you edit. This file has the name of the file you are working on and is given the extension .JOU. For example, if you start to edit a file named COLD.DAT, EDT creates a journal file named COLD.JOU.

EDT deletes the journal file when you make a normal exit from the editing session, unless you specify that it be saved. See Chapter 3 for more information.

Startup Command Files

A *startup command file* lets you customize the editor to meet your needs. You can use the startup command file to do the following automatically as you start EDT:

- Define keys
- Set options
- Define *macros*, which are sets of line editing commands that EDT executes when you enter the name of the macro
- Specify the appearance of text
- Define *entities of text*, which are character strings that EDT recognizes as a unit, such as a word or sentence

With a startup command file you could, for example, specify how many lines of a file EDT displays on your screen and how wide you want the display of text to be. You can even determine what functions you want certain keys to perform by redefining them.

EDTINI.EDT is the command file that EDT normally looks for when you start your editing session. EDTINI.EDT exists only if you create it.

The following is an example of an EDTINI.EDT file:

```
SET KEYPAD
SET WRAP 50
INCLUDE TEST.FIL
```

The commands in this EDTINI.EDT file cause EDT to prepare for keypad editing (SET KEYPAD), assume a right margin of 50 (SET WRAP 50), and put the contents of another file into the buffer ahead of the file contents you are about to edit (INCLUDE TEST.FIL).

You can name a startup file something other than EDTINI.EDT, especially if file EDTINI.EDT already exists. In this case, you must include a command file qualifier in the command line. A *command file qualifier* tells EDT to look for a specific file that contains startup commands. Both the EDTINI.EDT file and ways to specify other startup files are described in Chapter 4.

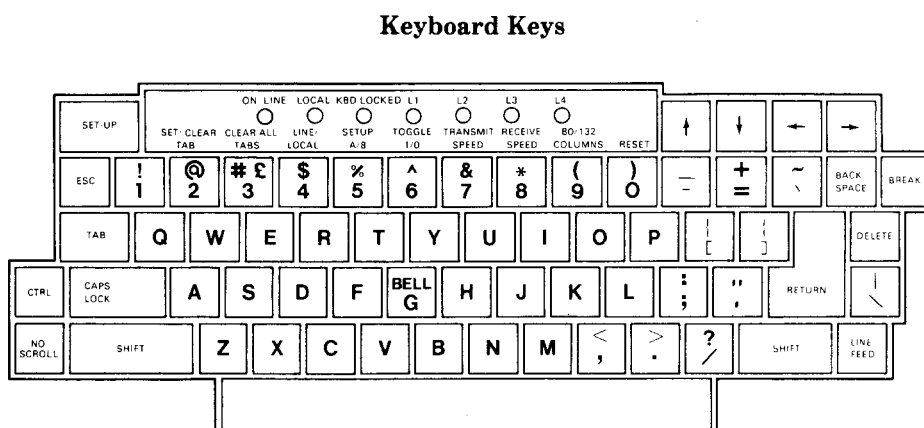
Keypad, Line, and Nokeypad Editing

With EDT you have the choice of using keypad, line, or nokeypad editing. Each of these ways of editing lets you:

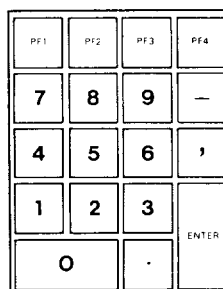
- Display a range of lines
- Locate, substitute, insert, and delete text
- Move and copy lines
- Copy text into a buffer and into other files

Keypad editing is available on VT52 or VT100 terminals. Figure 1-3 shows a VT100 keyboard with a keypad; the VT52 keyboard and keypad keys are similar. You press keys on the keypad to enter functions.

Figure 1-3: VT100 Keyboard and Keypad



Keypad Keys



Keypad editing is powerful and versatile, yet it is easy to learn and use. In keypad editing, the contents of a file are displayed on the screen as you edit. You can see the changes you make to a file as they take place.

There is a wide variety of keypad editing functions, each of which requires you to press only one or two keypad keys to perform the function. You enter commands, insert text, and perform CONTROL functions from the keyboard. CONTROL functions require you to hold down the `CTRL` key and press a keyboard key.

Line editing is useful to those who have hardcopy terminals or who prefer editing by numbered lines. In line editing, you make all entries from the keyboard. As you make changes to the contents of the file, EDT displays one or more lines at a time.

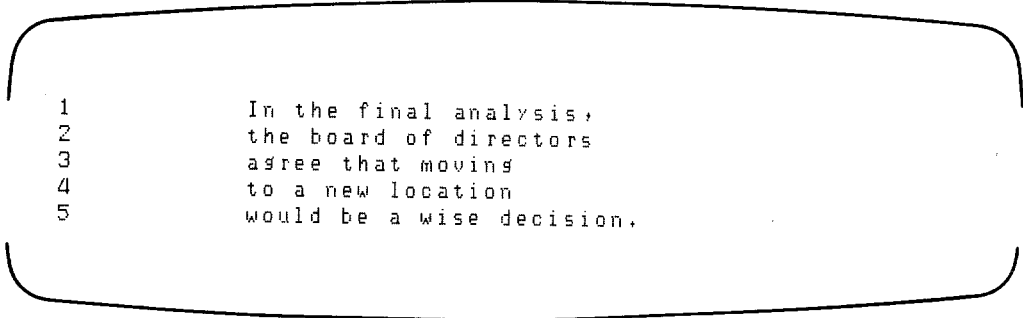
Nokeypad editing is useful if you have a terminal with a video screen but no keypad. With nokeypad editing, you enter all text and commands from the keyboard.

As with keypad editing, the file contents are displayed on the screen as you edit. However, you type in commands rather than press keys to make changes to the contents of the file. You also use nokeypad commands when you define keys.

Line Numbers

There are two types of line numbers in EDT: the line numbers that EDT assigns to the lines of text in a buffer and the *fixed* line numbers that a file may contain. Both kinds of line numbers are deleted when you end your editing session.

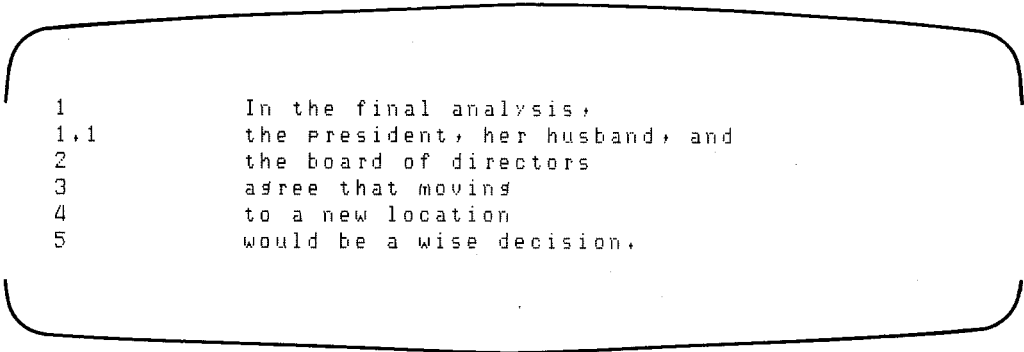
When you start EDT to work with the contents of a file, EDT automatically numbers each line. If you are creating a file, line numbers are assigned when you finish inserting text. These numbers are displayed during line editing but do not become part of your file. The numbering starts with 1 and is incremented by 1 for each line. The numbers are displayed at the left of the text:



```
1      In the final analysis,  
2      the board of directors  
3      agree that moving  
4      to a new location  
5      would be a wise decision.
```

In keypad and nokeypad editing, line numbers are assigned but are not displayed.

When you add a line between two other lines, EDT assigns this line a number between the numbers of the preceding and following lines:



```
1      In the final analysis,
1,1    the president, her husband, and
2      the board of directors
3      agree that moving
4      to a new location
5      would be a wise decision.
```

When there are no unassigned whole numbers between the preceding and following line, as shown above, the line you are adding is given a decimal number.

You can choose to store the contents of a buffer in VFC (variable with fixed control) files. You can assign *fixed line numbers* to these files.

You may find fixed line numbers useful when working with programs. EDT does not display fixed line numbers as you work on the contents of the file. However, EDT records these numbers internally. The lines are displayed after you have compiled the program.

You can use fixed line numbers in range specifications to locate lines. In addition, you can write fixed line numbers to a file. See Chapter 6 for more information.

Multiple Files and Buffers

Because of buffers, you can work with more than one file at a time. You can use the buffers available in EDT to:

- Move part or all of another file into your editing session
- Create a file from part or all of the text in a buffer
- Divide one or more files into sections

Buffers are extremely useful if you want to break one file into several files or combine several files or parts of files into one. See Chapter 7 for more information on editing with buffers.

Redefining Keys

You can redefine keys for keypad editing if you have a VT52 or VT100 terminal. The *DEFINE KEY* feature lets you customize the EDT editor to meet your needs. You can redefine any of the keypad keys and several of the CONTROL and keyboard keys. This feature lets you assign a series of nokeypad commands to a key. EDT performs these commands when you press the key. The *DEFINE KEY* feature is described in more detail in Chapter 10.

Types of EDT Commands

EDT commands, which are based on the English language, are divided into the following groups: the SET and SHOW commands, and keypad, line, and nokeypad editing commands.

SET and SHOW Commands

You can use SET and SHOW commands to control EDT editing functions (see Chapter 9). With these commands, you can:

- Set operating characteristics, such as the width of a display of text on the screen
- Show the status of selected operating characteristics

Keypad Editing Commands

Keypad editing commands are available on VT52 and VT100 terminals (see Chapter 5). In keypad editing, you press only one or two keypad keys to perform a number of common editing functions. Furthermore, in keypad editing it is possible to enter line editing and nokeypad editing commands. The ability to use these commands without leaving keypad editing increases the usefulness of keypad editing.

You can redefine the functions of the keypad keys and several of the keyboard keys (see Chapter 10). This feature lets you assign your own commands to a number of keys.

Line Editing Commands

Line editing commands operate on one or more lines of text (see Chapter 7). These commands, which are also available in both keypad and nokeypad editing, are most useful for manipulating large blocks of text. Users can extend this set of commands by defining macros, which are made up of one or more existing line mode commands.

Nokeypad Editing Commands

Nokeypad editing commands let you use all of the essential editing functions (see Chapter 8). These commands are most useful on terminals that have a video screen but not a keypad. You can use nokeypad editing on VT52 and VT100 terminals with the SET NOKEYPAD command. In addition, you can redefine keypad and CONTROL keys in terms of nokeypad commands.

Chapter 2

Sample Editing Sessions

This chapter gives you some general information about EDT editing and guides you through sample sessions of keypad, line and nokeypad editing. Red print shows what you enter. The functions printed in red on the keypad key diagrams show which function on the key is used.

See Chapters 5, 7, and 8 for more detailed descriptions of specific keypad, line, and nokeypad editing commands.

Getting Started

To use EDT, it is important that you understand the concept of buffers. You should also know the type of commands you need to begin an editing session, because the operating systems that support EDT do not all use the same command language.

Editing Text In Buffers

Buffers are essential to editing because they give you an area to work on text. When you start an editing session, EDT automatically provides a buffer that it names MAIN. If you edit an existing file, EDT puts a duplicate copy of the file's contents into the main buffer.

There are several kinds of buffers available when you edit text. EDT automatically creates a buffer named PASTE when you start an editing session. The paste buffer stores text from editing operations such as CUT. You can create other buffers to store text by naming them.

The text that you insert or revise in a buffer stays there until you save your edits (with the EXIT command) or delete them (with QUIT). If you save your edits, a new file is created. Existing versions of the file remain unchanged.

For example, suppose you start EDT and put some text for a memo into the main buffer. If you decide to save this draft of the memo, you end the editing session by typing the EXIT command. This command saves the contents of the buffer as a new file.

Now suppose you start EDT again to edit the file containing the memo. EDT places a copy of the memo into the main buffer. If you are dissatisfied with the edits you make during this editing session, enter the QUIT command. This deletes your edits and ends the editing session. The original draft of the memo remains unchanged.

You can return to the file containing the memo for as many editing sessions as you like. Each time you can either save or delete the buffer copy.

The Command Line

To start EDT, type the command line for your operating system after the system prompt. This prompt appears at the left of the screen when you log on. There may be a different prompt (such as \$, DCL >, or Ready) for each operating system that has EDT. Therefore, this manual represents all operating system prompts with the following symbol:



The operating system you use determines which one of the following CLIs (Command Line Interpreters) you need to start EDT:

- DCL (Digital Command Language) or PDS (Program Development System)
- MCR (Monitor Console Routine) or CCL (Concise Command Language)

The DCL and PDS command languages use one command line; the MCR and CCL languages use another. The examples in this manual show both the DCL/PDS and MCR/CCL command lines.

Table 2-1 lists the CLIs and command level prompts for PDP-11 and VAX/VMS operating systems. You will need this information to use the DCL/PDS or MCR/CCL command line formats.

Table 2-1: Operating System CLIs and Prompts

| Operating System | CLI | Prompt |
|------------------|-----|---------------------------|
| IAS | PDS | PDS > or SCI > |
| | MCR | MCR > or PDS >> or SCI >> |
| VAX/VMS | DCL | \$ |
| RSTS/E | CCL | Ready |
| RSX-11M | MCR | > |
| RSX-11M-PLUS | MCR | > |
| | DCL | DCL > |

Throughout the manual, the choice of command lines will be as shown:

DCL/PDS

 **DCL/PDS**

MCR/CCL


 **MCR/CCL**

The examples in this manual are designed so you can follow along at your terminal.

Starting an EDT Session

Start EDT by typing EDIT/EDT or EDT after the prompt, depending on your system's command language. Both DCL/PDS and MCR/CCL command lines are shown. When you create a file, you can assign any file name and extension or file type that your system allows; check your system documentation for any limitations. In this example, enter the file name "COLD.DAT" and press **RET**:

DCL/PDS

```
 EDIT/EDTRET
_File: COLD.DATRET
Input file does not exist
[EOB]
*
```

NOTE

Some systems may use a question mark (File?) instead of a colon (File:) in prompting you for a file name.

MCR/CCL

```
EDT(RET)
EDT> COLD.DAT(RET)
Input file does not exist
[EOB]
*
```

Pressing (RET) completes your commands. The system looks for a file with the name you gave. The message "Input file does not exist" means that the file has not yet been created. Any text that you now insert into the buffer appears before the [EOB] symbol. The asterisk (*) is the line editing prompt, which lets you know you started EDT.

Keypad Editing

You will probably want to use keypad editing if you have a VT52 or VT100 terminal. Keypad editing is:

- Convenient — You can use most editing functions by pressing one or two keypad keys.
- Fast — You press keys rather than type commands, so you save time and avoid typographical errors.
- Versatile — You can customize the editor to meet your needs by redefining keys.

The keypad is located at the right of the keyboard. Figures 2-1 and 2-2 show diagrams of the VT100 and VT52 keypads.

The keypad function keys used in the examples are enclosed by broken lines. For example:

GOLD

NOTE

GOLD is a *function* in keypad editing. The actual key is blue on the VT52 keypad. On VT100s, the GOLD key is labeled PF1.

The examples in this section show most of the keypad editing functions. If you do not have a VT100 or VT52 terminal, you may want to skip this section and read the descriptions of line editing or nokeypad editing instead. (If you are following the examples at your terminal and plan to skip the keypad editing section, type QUIT after the asterisk and press **(RET)** .)

Starting Keypad Editing

Type the CHANGE command after the line editing prompt to start a keypad editing session:

* CHANGE**(RET)**

The screen clears. The cursor and the [EOB] symbol appear at the upper left of the screen. Whatever you type at this point is displayed on the screen before the cursor. The [EOB] symbol always appears after the last character in the text buffer.

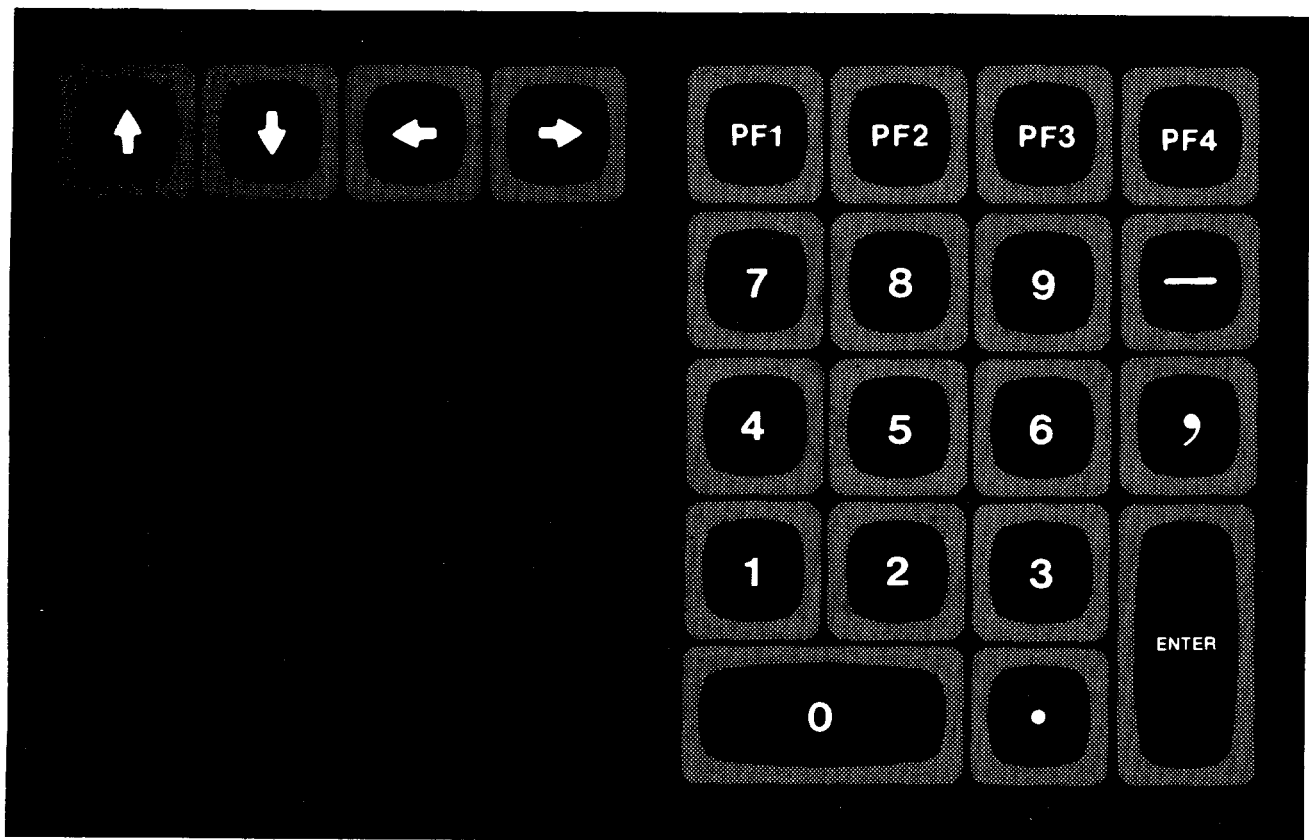
The HELP Facility

You can get two levels of help descriptions in keypad editing. First, you can press the HELP key to get a diagram of the default keypad functions. (Refer to Figures 2-1 and 2-2.) The screen goes blank for a few seconds after you press the HELP key, and then a diagram of keypad functions appears. (Requesting help does not affect your editing session; pressing the space bar returns you to where you left off.)

Second, after EDT displays the diagram, you can press any keypad key to see a description of what the key does. If you press the keypad key numbered 5, for example, a description of the BACKUP and TOP functions appears on the screen. (BACKUP and TOP are the functions associated with keypad key 5.) At the bottom of each description are directions for seeing the whole keypad diagram again (by pressing **(RET)**), reading about other functions (by pressing another keypad key), or returning to your editing session (by pressing the space bar).

If you redefine the keyboard or keypad keys, the HELP diagrams do not reflect this fact.

Figure 2-1: VT100 Keypad Functions



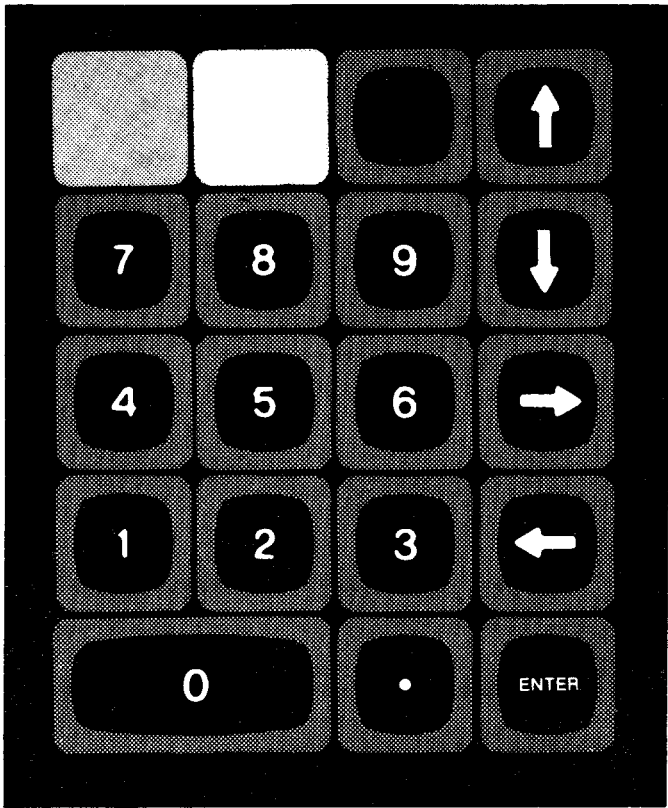
A. The Keypad and the Arrow Keys

| | | | |
|----|------|------|-------|
| UP | DOWN | LEFT | RIGHT |
|----|------|------|-------|

| | | | |
|-------------------|----------------|-------------------|----------------|
| GOLD | HELP | FNDNXT FIND | DEL L UND L |
| PAGE COMMAND | SECT FILL | APPEND REPLACE | DEL W UND W |
| ADVANCE BOTTOM | BACKUP TOP | CUT PASTE | DEL C UND C |
| WORD CHNGCASE | EOL DEL EOL | CHAR SPECINS | ENTER SUBS |
| LINE OPEN LINE | | SELECT RESET | |

B. Keypad Functions

Figure 2-2: VT52 Keypad Functions



A. The Keypad

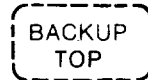
| | | | |
|-------------------|----------------|-----------------|------------------|
| GOLD | HELP | DEL L UND L | UP REPLACE |
| PAGE COMMAND | FNDNXT FIND | DEL W UND W | DOWN SECT |
| ADVANCE BOTTOM | BACKUP TOP | DEL C UND C | RIGHT SPECINS |
| WORD CHNGCASE | EOL DEL EOL | CUT PASTE | LEFT APPEND |
| LINE OPEN LINE | | SELECT RESET | ENTER SUBS |

B. Keypad Functions

Keypad Functions

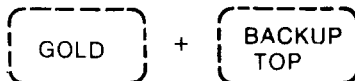
Most of the keypad keys have both a standard and an alternate function. You use the *standard function* (the upper of two functions) by pressing the key. Pressing the GOLD key first causes EDT to perform the *alternate function*.

In this manual, a statement such as "Use the BACKUP function" means that you press the BACKUP function key:



BACKUP is the standard function of the key.

The statement "Use the TOP function" means that you press GOLD and then the TOP function key:



TOP is the alternate function.

If you want to follow along with the examples in this section, press the keypad labeled 4 to set operations to ADVANCE:



Inserting Text

You can enter text as soon as you start keypad editing. Whatever you type on the keyboard is inserted directly into the buffer. The following paragraph appears on your screen as you type it.

The town of Cold, Montana is located on a butte.(RET)
The average annual snowfall is 175 inches.(RET)
Cold has a very active ski resort and several(RET)
supporting businesses: a ski shop, two(RET)
restaurants, and three stores.(RET)
[EOB]

The cursor is positioned after the last character you enter.

Moving The Cursor

You can use the TOP function to move the cursor to the top of the buffer:

[GOLD] + [BACKUP
TOP]

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
supporting businesses: a ski shop, two
restaurants, and three stores.
[EOB]

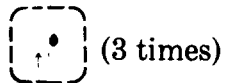
You can move the cursor to the bottom of the buffer with the BOTTOM function:

[GOLD] + [ADVANCE
BOTTOM]

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
supporting businesses: a ski shop, two
restaurants, and three stores.

[EOB]

The arrow keys on the keyboard let you move the cursor in each of four directions:



The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
supporting businesses: a ski shop, two
restaurants, and three stores.

[EOB]



The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
supporting businesses: a ski shop, two
restaurants, and three stores.

[EOB]

→ (4 times)

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
supporting businesses: a ski shop, two
restaurants, and three stores.

[EOB]

→ (2 times)

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
supporting businesses: a ski shop, two
restaurants, and three stores.

[EOB]

With some keys you can move the cursor by characters, words, and lines. For example, the WORD function key moves the cursor to the next word:

WORD
CHNGCASE

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
supporting businesses: a ski shop, two
restaurants, and three stores.
[EOB]

Using Select Ranges to Move Text

This section shows how to:

- Mark text with the SELECT function
- Delete the selected text with the CUT function
- Move the cursor to another location in the buffer
- Reinsert the deleted text before the cursor with the PASTE function

You can mark text with select ranges for several keypad operations. When you move the cursor by pressing an arrow key or a function key such as LINE or WORD, you determine the text to be contained in the select range. The select range remains until you press another keypad function key, such as CUT.

The examples in this section show VT100 select ranges, since the reverse video feature makes select ranges obvious. Characters in *reverse video* are either black with a white background or white with a black background, in contrast with the rest of the characters on the screen. The VT52 also has the select range function, although it does not use reverse video.

The following series of frames shows what happens when you mark a select range and delete the selected text with the CUT function. The text is then reinserted with the PASTE function.

1. Move the cursor to the start of the text that you want to cut (the first "r" in the word "restaurant"). Press the SELECT function key, and then press the WORD function key three times:

 (13 times) +  +  (3 times)

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
supporting businesses: a ski shop, two
restaurants, and three stores.

[EOB]

(If you press the SELECT function key at the wrong place, press GOLD and the RESET function key and try again.)

2. Press the CUT function key to delete the text in the SELECT range:

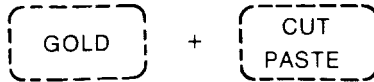


The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
supporting businesses: a ski shop, two
stores.

[EOB]

The text that you delete with the CUT function is stored in the paste buffer. The deleted text remains in the paste buffer until the next CUT operation.

3. You can reinsert the deleted text with the PASTE function one or more times. To reinsert the text at the original location, press the GOLD and PASTE function keys:



The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
supporting businesses: a ski shop, two
restaurants, and three stores.

[EOB]

When you press the CUT function key in this example, you delete text and therefore cancel the select range. If you now press CUT, EDT beeps and displays the following message at the bottom left of the screen:

No select range active

Deleting Words and Characters

You can delete words and characters several ways. The following examples use the DELETE and DEL W functions.

Press the LEFT arrow function key seven times to move the cursor to the space after "and". Then press the DELETE key twice to delete characters preceding the cursor:



For the purpose of this example, retype the characters "nd" and press the RIGHT arrow function key once so the line reads:

restaurants, and three stores.

To delete the word that the cursor is on, use the DEL W function:

DEL W
UND W

restaurants, and stores.

You can reinsert the last word you deleted with the UND W function:

GOLD + DEL W
UND W

restaurants, and three stores.

Deleting Lines

There are three keypad functions (DEL L, DEL EOL, and UND L) that let you delete and undelete lines. When you delete a line with DEL L or DEL EOL, the line is stored in the line buffer until you replace it with another deleted line. UND L reinserts the line that you deleted.

The DEL L function lets you delete a line, including the line terminator. In this example, move the cursor to the beginning of the fourth line:

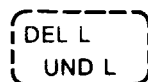
LINE
OPEN LINE

⋄ (2 times)

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
supporting businesses: a ski shop, two
restaurants, and three stores.

[EOB]

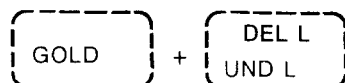
Press the DEL L function key. The text following the deleted line moves next to the cursor.



The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
restaurants, and three stores.

[EOB]

Undelete the line with UND L:



The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
supporting businesses: a ski shop, two
restaurants, and three stores.

[EOB]

Delete all or part of a line with the DEL EOL (Delete to End of Line) function. This deletes all of the text from the cursor to the following line terminator. The text following the deletion does not move:

GOLD + EOL
DEL EOL

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
restaurants, and three stores.

[EOB]

You can undelete the line with UND L:

GOLD + DEL L
UND L

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
supporting businesses: a ski shop, two
restaurants, and three stores.

[EOB]

How to Change Case

You can change the case of text by individual characters or by whole strings of characters. Move the cursor to the letter "o" in "Cold" in the third line:




The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
supporting businesses: a ski shop, two
restaurants, and three stores.

[EOB]

Change the case of one character at a time by pressing the GOLD and CHNGCASE key sequence three times:

Cold +  +  (Press GOLD and CHNGCASE three times in succession) COLD

If you use a select range with CHNGCASE, everything within the range changes case. Move the cursor up to the word "town":

 (2 times)

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
COLD has a very active ski resort and several
supporting businesses: a ski shop, two
restaurants, and three stores.

[EOB]

Press SELECT, then press the WORD function key four times; this creates a select range:



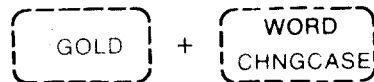
The town of Cold, Montana is located on a butte.

When you press the GOLD and CHNGCASE function keys, the characters you select change case.



The TOWN OF cOLD, mONTANA is located on a butte.

When you press GOLD and CHNGCASE to change the case of the characters in a select range, the characters change case and the select range is canceled. If you then press GOLD and CHNGCASE again, EDT changes the case of the character to the right of the cursor:



The TOWN OF cold, mONTANA Is located on a butte.

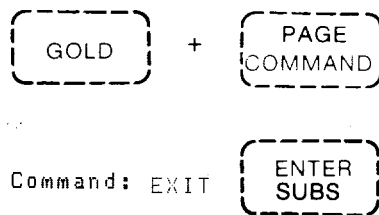
Ending a Keypad Editing Session

There are two ways to exit keypad editing:

1. Press **CTRL/Z**, which returns you to line editing. (To use **CTRL/Z**, hold down the CONTROL key and press the keyboard key labeled Z.) You still need to type EXIT or QUIT after the asterisk prompt to leave EDT.

If instead you type CHANGE after the asterisk prompt and press **RET**, EDT returns to keypad editing. The cursor returns to where it was when you pressed **CTRL/Z**.

2. Use the **COMMAND** function and type **EXIT** (to save your edits in a file) or **QUIT** (to exit without saving your edits). Then press **ENTER**:



Chapter 5 gives you more information about editing with keypad functions. It describes not only the basics presented here, but also different ways of using each function.

If you will be using only keypad editing, skip the following sections and proceed to Chapter 3.

Line Editing

Line editing is commonly used on hardcopy terminals. It allows you to edit several lines of text at once, and to manipulate text in different buffers. In addition, line editing is useful on video terminals if you prefer to edit by numbered lines.

You can abbreviate most of the line editing commands, although the entire command is given in these examples. Pressing **(RET)** completes the commands.

Chapter 7 is provided as a reference source for line editing commands and their abbreviations.

Creating Files

When you start EDT and give a file name, EDT automatically provides buffer space and prepares for line editing. (You can change this default with a startup command file, which is described in Chapter 4.) The asterisk prompt (*) shows that you can enter line editing commands.

The simplest way to create a file is to insert new text. This section uses the following line editing commands to create a file and enter text:

- **INSERT**
- **TYPE**
- **Null**
- **EXIT**

It does not matter whether you enter these commands in uppercase or lowercase type, although uppercase is used in the examples.

The line editing asterisk prompt (*) appears when you start EDT. Enter the following command lines:

DCL/PDS

```
␣ EDIT/EDT␣  
_File: FALLS.NEW␣  
Input file does not exist  
[EOB]  
*␣
```

MCR/CCL

```
␣ EDT␣  
EDT> FALLS.NEW␣  
Input file does not exist  
[EOB]  
*␣
```

Now you can insert the contents of a file that will be named FALLS.NEW.

The INSERT command lets you put text into the main buffer. When you insert text, each line is indented 16 spaces to allow for line numbering to the left. Indenting is on the screen only and does not become part of the edited file.

Enter the command and text as shown, ending each line with **(RET)**. When you finish entering text, press **(CTRL/Z)**. (On your terminal, the **(CTRL/Z)** appears as **^Z**.)

```
*INSERT(RET)
```

```
Quahasset's good schools and many(RET)  
recreational opportunities make it(RET)  
attractive to families. In addition,(RET)  
its tax rates are reasonable and it(RET)  
can support new housing without(RET)  
overcrowding.(RET)  
(CTRL Z)
```

```
* (RET)
```

As you enter a line, you can delete characters by pressing **(DEL)**. You can delete a portion of a line, from the cursor to the left margin, by entering **(CTRL/U)**. **(CTRL/Z)** returns you to the line editing prompt. However, when you end a line with **(RET)**, you can no longer delete characters from that line in insert mode.

Enter **TYPE WHOLE** to display the contents of the file:

```
* TYPE WHOLE(RET)
```

```
1 Quahasset's good schools and many  
2 recreational opportunities make it  
3 attractive to families. In addition,  
4 its tax rates are reasonable and it  
5 can support new housing without  
6 overcrowding.  
[EOB]
```

```
* (RET)
```

Notice that EDT numbers each line.

WHOLE is only one of the entities you can use with the TYPE command. Line numbers are examples of other entities. You can display individual lines by entering TYPE and then a line number. To display consecutive lines, enter the Null (implied TYPE) command by pressing (RET) to display consecutive lines:

```
* TYPE 1(RET)
  1      Quahasset's good schools and many
* (RET)
  2      recreational opportunities make it
* (RET)
  3      attractive to families. In addition,
* (RET)
```

You can use a *range specification* to display a series of lines. Enter the TYPE command and two line numbers separated by a colon (:):

```
* TYPE 2:5(RET)
  2      recreational opportunities make it
  3      attractive to families. In addition,
  4      its tax rates are reasonable and it
  5      can support new housing without
* (RET)
```

To enter text between lines 3 and 4, first mark your position by displaying the fourth line:

```
* (RET)
  4      its tax rates are reasonable and it
* (RET)
```

Then type INSERT after the asterisk and press **RET**. Enter the line and press **CTRL/Z** to complete your insertion:

```
* INSERTRET
our researchers strongly feel thatRET
CTRL Z
*
```

To check the insertion, display the file again with TYPE WHOLE:

```
* TYPE WHOLERET
1      Quahasset's good schools and many
2      recreational opportunities make it
3      attractive to families. In addition,
3.1    our researchers strongly feel that
4      its tax rates are reasonable and it
5      can support new housing without
6      overcrowding.
[EOB]
*
```

If you want to save the file and end the editing session, type EXIT after the asterisk:

```
* EXITRET
```

When you type EXIT and press **RET**, EDT moves the contents of the text buffer to a file and names the file FALLS.NEW. (You assigned the file name FALLS.NEW when you entered the command line to start EDT.) When you exit, EDT displays a message stating that your file is saved and tells you the number of lines of text it contains. The exact wording of the message varies from system to system. The operating system prompt appears on the line after the message.

Using Line Editing Commands

Editing sessions are usually more complex than the preceding session. The next example shows how to use more of the line editing commands. The commands used are:

- INSERT
- REPLACE
- INSERT END
- TYPE
- EXIT

For the following example, start EDT and create the file CHILLY.DAT:

DCL/PDS

```
⌘ EDIT/EDT CHILLY.DAT (RET)
Input file does not exist
[EOB]
* █
```

MCR/CCL

```
⌘ EDT CHILLY.DAT (RET)
Input file does not exist
[EOB]
* █
```

Type INSERT and press (RET) after the asterisk prompt and enter text into the buffer. Complete your insertion by pressing (CTRL/Z):

```
*INSERT(RET)
The town of Chilly, Me. is located 40(RET)
miles north of Bangor.(RET)
(RET)
Chilly's main industry is tourism and(RET)
it now contains no heavy or light(RET)
industries. There is a sufficient(RET)
labor base for a small light industry.(RET)
but much of the labor force is unskilled.(RET)
(RET)
The average yearly temperature is 68(RET)
degrees F; mean annual snowfall is(RET)
172 inches.(RET)
(CTRL/Z)
*
```

In order to use many of the line editing operations, you need to locate the line where a command is to take effect. There are several ways to find a line in the text buffer. One method is to make a best guess on the line number and then search from that point.

Suppose you want to add a statement after the second line. To locate the position for adding the line, enter TYPE 2 and press (RET). EDT displays line 2 and the line editing prompt.

```
*TYPE 2(RET)
2 miles north of Bangor.
*
```

Enter **RET** to display the next line (the blank line 3) in the text buffer:

```
* RET
  3
* RET
```

You can insert a statement in place of line 3 with the REPLACE command. When you type REPLACE 3 and press **RET**, EDT deletes line 3 and displays a message telling how many lines were deleted. Then EDT enters insert mode. Now you enter two lines of text and a blank line. Press **CTRL/Z** to complete the insertion:

```
* REPLACE 3RET
  1 line deleted
```

```
Major access is by route 70/81,RET
which is frequently snowbound.RET
RET
CTRL/Z
```

```
* RET
```

Assume that you want to add a recommendation to the report. To insert the recommendation at the end of the buffer, use the INSERT END command. EDT sets a position in the text buffer so that the text you insert is appended to the end of the buffer:

```
* INSERT ENDRET
```

```
RET
Because of its inaccessibility and itsRET
lack of a skilled labor force, we do notRET
recommend Chilly, Me. as a plant site.RET
CTRL/Z
```

```
* RET
```

Enter the TYPE WHOLE command to review the contents of the text buffer. EDT displays each line of the text buffer in sequence, in a process called *scrolling*, beginning with the first line. You can interrupt the scrolling action with **CTRL/S**. To restart the scrolling action, enter a **CTRL/Q**. Notice that the lines you inserted between lines 2 and 3 are numbered with decimal numbers.

```
* TYPE WHOLE(RET)
1 The town of Chilly, Me. is located 40
2 miles north of Bangor.
2.1 Major access is by route 70/81,
2.2 which is frequently snowbound.
2.3
4 Chilly's main industry is tourism and
5 it now contains no heavy or light
6 industries. There is a sufficient
7 labor base for a small light industry,
8 but much of the labor force is unskilled.
9
10 The average yearly temperature is 68
11 degrees F; mean annual snowfall is
12 172 inches.
13
14 Because of its inaccessibility and its
15 lack of a skilled labor base, we do not
16 recommend Chilly, Me. as a plant site.
[EOB]
*
```

Now type EXIT and press **RET**:

```
* EXIT(RET)
```

The EXIT command moves the contents of the text buffer to a file and ends the editing session. EDT then displays a message with the complete name of the file (CHILLY.DAT) and the number of lines in the file. The wording of this message varies from system to system. The operating system prompt appears on the next line.

Revising Files

Editing existing files is quite similar to editing new files. When the file name you enter in your command line specifies an existing file, EDT puts the contents of this file into the text buffer. You can enter the same commands that you would for a new file after the line editing prompt.

For the next example, assume that you want to add to the CHILLY.DAT file. The EDT commands used are:

- INSERT
- TYPE
- SUBSTITUTE
- DELETE
- RESEQUENCE

Start EDT to edit the file CHILLY.DAT. When EDT puts the contents of the file into the text buffer, it displays the first line of the file and a line editing prompt:

DCL/PDS

EDIT/EDT CHILLY.DAT^{RET}

MCR/CCL

EDT CHILLY.DAT^{RET}

```
1          The town of Chilly, Me. is located 40
*  █
```

Assume you want to insert some text before line 5. Type in the following command and text and press ^{CTRL/Z} :

```
* INSERT 5RET
The nearest airport is in Bangor,RET
although a seaplane lands once a weekRET
on nearby Lake Okichibwa.RET
CTRL/Z
*  █
```

Now you decide to replace the highway number in the first paragraph with another string of characters. You use the SUBSTITUTE command to replace the search string (the highway number 70/81) with a substitute string (another highway number, 193). The *strings* in this example are the characters that EDT locates and replaces. You need *delimiters* to separate these strings.

The slash (/) is one of the special characters frequently used to delimit strings. However, note that there is a slash within the search string "70/81". Therefore, you can use the exclamation point (!) as a delimiter in this example. (You could use any of the special characters, such as & or \$.)

Locate the line containing the search string "70/81" with the TYPE command. Then make the substitution as shown. When EDT completes the substitution, it shows the corrected line and a message about the number of substitutions made.

```
* TYPE 2(RET)
  2          miles north of Bangor.
* (RET)
  3          Major access is by route 70/81,
* SUBSTITUTE!70/81!193!(RET)
  3          Major access is by route 193,
1 substitution
* █
```

You then decide to delete the lines that discuss temperature and snowfall. Use the TYPE command to display the lines and the DELETE command to delete them. Use THRU to delete consecutive lines. EDT deletes the lines, tells you how many lines were deleted, and displays the next line.

```
* TYPE 12(RET)
  12          The average yearly temperature is 68
* DELETE 12(RET)
1 line deleted
  13          degrees F; mean annual snowfall is
* DELETE 13 THRU 14(RET)
2 lines deleted
  15
* █
```

If you enter DELETE without a line number, EDT deletes the current line.

Suppose you want to insert a final statement. Type INSERT END and press (RET) to enter this statement at the end of the buffer, and complete the insert with (CTRL/Z):

```
* INSERT END(RET)
                                (RET)
                                Factors considered include access,(RET)
                                present industry, and labor force.(RET)
                                (CTRL Z)
*
```

To change the decimal line numbers to whole numbers, enter the RESEQUENCE command. EDT renumbers the lines from 1 through 21. Enter a TYPE WHOLE command to review the contents of the text buffer:

```
* RESEQUENCE(RET)
21 lines resequenced
* TYPE WHOLE(RET)
  1          The town of Chilly, Me. is located 40
  .
  .
  .
  21         present industry, and labor force.
[EOB]
*
```

Type the EXIT command and press (RET) to end the editing session:

```
* EXIT(RET)
```

The EXIT command moves the contents of the text buffer to a file. EDT then displays a message with the complete name of the file and the number of lines in the file.

The line numbers are deleted when you exit. However, when you start an editing session EDT rennumbers each line of the buffer, beginning with 1 and continuing in increments of 1.

To practice working with files, create two new files similar to CHILLY.DAT and insert about ten lines of text in each. You will need these files for the next example. Name the files TEPID.DAT and WARM.DAT. The steps for creating these files are:

1. Type the command line to start EDT and enter one of the two file names (TEPID.DAT or WARM.DAT).
2. Use the INSERT command to enter a few lines, then press **CTRL/Z** to complete the insertion.
3. Save each of the new files with the EXIT command.

Multiple Files and Buffers

When you start an editing session, EDT places the text you work with into the main text buffer, unless you specify otherwise. At the end of the session, EDT moves the contents of the main text buffer to a file. You can, however, create several text buffers during an editing session and save or delete their contents. You create a buffer simply by assigning a name to it.

During one editing session, you can put several files into one or more buffers and generate several output files. The next example uses the following commands:

- INSERT
- INCLUDE
- SHOW
- COPY
- MOVE
- RESEQUENCE
- WRITE
- EXIT

DCL/PDS

MCR/CCL

In the following example, assume you write a report that uses information from the other files you have created. Enter the **INSERT** command after the asterisk and type the following:

*

Now use the INCLUDE command to bring a copy of your existing files (CHILLY.DAT, WARM.DAT, and TEPID.DAT) into this editing session. The INCLUDE command brings copies of these files into your editing session in separate text buffers. (Because you are using only copies of these files, the originals remain unchanged.)

```
* INCLUDE CHILLY.DAT =CHILLY(RET)  
* INCLUDE WARM.DAT =WARM(RET)  
* INCLUDE TEPID.DAT =TEPID(RET)  
*
```

Note that you created the three buffers simply by naming them.

Use the SHOW BUFFER command to display the names of these new text buffers. The equal sign (=) indicates the current text buffer.

```
* SHOW BUFFER(RET)  
=TEPID 10      lines  
WARM   10      lines  
CHILLY 21      lines  
MAIN   12      lines  
PASTE  0       lines  
*
```

Type =CHILLY and press **(RET)** to make CHILLY the current buffer. The *current* buffer is the buffer in which your commands take effect. EDT displays the contents of the text buffer CHILLY.

```
* =CHILLY(RET)
1      The town of Chilly, Me. is located 40
:
:
:
21     Present industry, and labor force.
[EOB]
*
```

Now use the COPY command to copy lines 20 and 21 of the text buffer CHILLY to the end of the main text buffer. The cursor location moves to the line just after the copied text in the main buffer.

```
* COPY =CHILLY 20,21 TO =MAIN END(RET)
2 lines copied
*
```

Use the MOVE command to move the contents of the text buffers to the end of the main text buffer:

```
* MOVE =CHILLY TO =MAIN END(RET)
21 lines moved
* MOVE =WARM TO END(RET)
10 lines moved
* MOVE =TEPID TO END(RET)
10 lines moved
* *
```

MAIN now includes the title of the report, the introductory material of the report, the text from CHILLY.DAT, WARM.DAT, and TEPID.DAT, and the conclusion copied from lines 20 and 21 in CHILLY.DAT.

Now renumber the lines in the main buffer with the RESEQUENCE command. (Enter TYPE WHOLE if you want to read the file before saving it.) Then use the WRITE command to copy the contents of the main buffer to the new file named SITES.SUM. When you complete the EXIT command, you will have a file named SITES.SUM and a file named SITES.DAT.



```
* RESEQUENCE(RET)
55 lines resequenced
* WRITE SITES.SUM(RET)
```

The system then displays a message that tells you the number of lines in the file. Type the EXIT command to save the file:

```
* EXIT(RET)
```

Nokeypad Editing

With nokeypad editing you can display whole sections of text on a video screen. Therefore, you can see how the edits you make affect the contents of a buffer.

Nokeypad editing has essentially the same capabilities as keypad editing, although they are two distinct kinds of editing. Whereas you press keys to use most functions in keypad editing, you must type in nokeypad editing commands.

Chapter 8 is provided as a reference source of nokeypad commands.

Starting Nokeypad Editing

VT52 and VT100 terminals normally enter keypad editing when you use the CHANGE command. If you have a VT52 or VT100, start nokeypad editing from line editing by typing SET NOKEYPAD after the asterisk and pressing (RET). On other video terminals, nokeypad is the default mode, in which case only the CHANGE command is necessary.

The SET NOKEYPAD command readies EDT for nokeypad editing. The screen clears when you enter the CHANGE command. Then EDT displays the contents of the text buffer or a blank screen if you are creating a file. The cursor initially appears in the upper left of the screen.

In nokeypad editing, the screen displays up to 22 lines of text. No line numbers are shown. When you type the first character of a command, the cursor jumps to the bottom left of the screen. The commands that you type are displayed at the lower left of the screen until you complete them by pressing (RET).

You can move the cursor to any position in the display of text with cursor movement commands. The format for a cursor movement command is a number followed by an entity (such as 3L for 3 lines), ending with (RET).

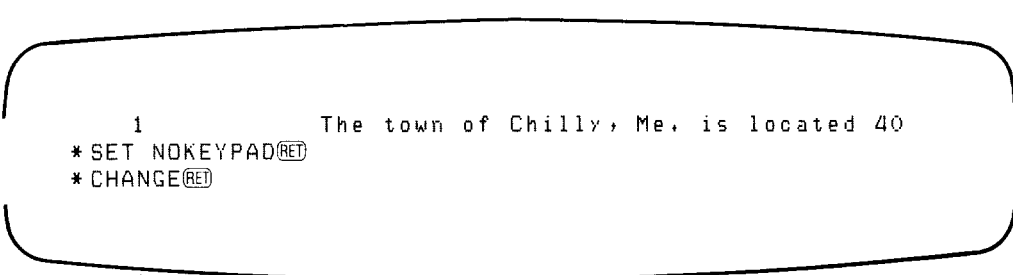
Start EDT, then enter the SET NOKEYPAD and CHANGE commands as shown:

DCL/PDS

EDIT/EDT CHILLY.DAT(RET)

MCR/CCL

EDT CHILLY.DAT(RET)



```
1          The town of Chilly, Me. is located 40
* SET NOKEYPAD(RET)
* CHANGE(RET)
```

The screen clears, and then the contents of CHILLY.DAT are displayed starting at the upper left:

```
The town of Chilly, Me. is located 40
miles north of Bangor.
Major access is by route 193,
which is frequently snowbound.
```

```
.
```

```
Factors considered include access,
present industry, and labor force.
[EOB]
```

Nokeypad Editing Commands

In nokeypad editing, the cursor resides in the display of text until you type a command, which appears at the bottom left of the screen. When you press (RET) to execute the command, the cursor returns to the display of text.

The examples in this section show how to:

- Move the cursor
- Insert text
- Delete and replace text
- Substitute text
- Cut and paste text
- End the nokeypad session

The first character that you type in nokeypad editing causes the cursor to jump to the bottom left of the screen. In this example, start the session by moving the cursor down three lines with the "3L" command. Watch where the cursor moves when you type the number "3":

```
The town of Chilly, Me. is located 40
miles north of Bangor.
Major access is by route 193,
which is frequently snowbound.
```

```

.
.
Factors considered include access,
present industry, and labor force,
[EOB]
```

```
3
```

When you finish typing "3L" and then press **(RET)**, the cursor moves to the first character in the fourth line:

```
3L(RET)
```

The text appears as follows:

```
The town of Chilly, Me. is located 40
miles north of Bangor.
Major access is by route 193,
which is frequently snowbound.
```

```

.
.
Factors considered include access,
present industry, and labor force,
[EOB]
```

Assume you decide to shorten the file. Move the cursor to the beginning of the eighth line by typing 4L and pressing **(RET)**:

```
4L(RET)
```

The cursor moves to the beginning of the eighth line. Delete the rest of the lines in the buffer by typing D14L (delete 14 lines) and pressing **RET** :

```
D14L RET
```

Because you deleted the rest of the lines in the buffer, the cursor is now at the end:

```

      .
      .
      .
on nearby Lake Okichibwa.
[EOB]
```

“Okichibwa” should be spelled “Okechibwa.” Use the following command sequence to move the cursor:

```
-1L RET
17C RET
```

The cursor moves one line up and 17 characters (17C) to the right. The minus sign is necessary in the first command to set the direction of cursor movement backwards. The cursor is now on the letter “i”:

```
on nearby Lake Okichibwa.
[EOB]
```

To delete the incorrect letter and insert the letter “e”, use the following command sequence:

```
D1C RET
Ie CTRL/Z RET
```

These commands delete the letter “i” and insert the letter “e”. **CTRL/Z** completes the insertion and **RET** executes the command. The line now reads:

```
on nearby Lake Okechibwa.
```


The name of the lake is actually "Okechibwa Lake." Move the cursor to the beginning of the word "Lake" with the command —8C:

-8C(RET)

The cursor moves to the "L" in "Lake":

on nearby Lake Okechibwa.

Store the word "Lake" in the word buffer with the command D1W, which deletes one word:

D1W(RET)

"Lake" disappears from the screen:

on nearby Okechibwa.

Move the cursor 9 spaces to the right with the command 9C:

9C(RET)

The cursor position is on the period:

on nearby Okechibwa.

Now you can reinsert the word "Lake" by typing the undelete word command:

UNDW(RET)

The line reads:

on nearby OkechibwaLake .

To enter a space between the two words, type "I", press the space bar once, and then press CTRL/Z and (RET) :

I SP(CTRL/Z) (RET)

Move the cursor to the space before the period by typing "4C" and pressing **(RET)**, and then enter the delete character command:

```
4C(RET)  
D1C(RET)
```

The result is:

```
on nearby Okechibwa Lake.
```

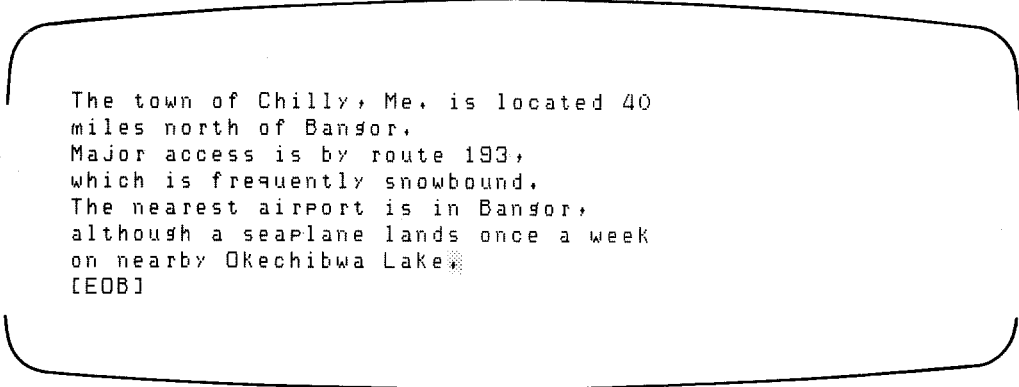
If you decide to change the word "frequently" to "often" in the fourth line, you can use the substitute command:

```
S/frequently/often/(RET)
```

However, when you try this, you get the error message:

```
String was not found
```

To understand why this happened, look at the cursor position in the text area:



```
The town of Chilly, Me. is located 40  
miles north of Bangor.  
Major access is by route 193,  
which is frequently snowbound.  
The nearest airport is in Bangor,  
although a seaplane lands once a week  
on nearby Okechibwa Lake.  
[EOB]
```

The substitute command began a forward search when the cursor was near the end of the text and immediately reached the end of the buffer. To reverse the search direction, type "-S" when you substitute the strings:

```
-S/frequently/often/(RET)
```

This time the substitution is made:

which is often snowbound.

EDT will do a reverse search the next time unless you negate the minus, also known as BACK, with ADV (ADVANCE). Type ADV and press (RET) to reset all searches to forward:

ADV(RET)

You now decide to perform one last edit on this text. You want to move the second sentence to the end of the buffer. Move the cursor to the beginning of the second sentence by typing -2L:

-2L(RET)

The cursor moves to the location:

Major access is by route 193,
which is often snowbound.

Type CUTSEN to cut the sentence and move it to the PASTE buffer:

CUTSEN(RET)

Your text now reads:

The town of Chilly, Me. is located 40
miles north of Bangor.
The nearest airport is in Bangor,
although a seaplane lands once a week
on nearby Okechibwa Lake.
[EOB]

To reinsert the deleted sentence at the end of the buffer, move the cursor to the end by typing 3L and (RET), and then type the PASTE command:

3L(RET)
PASTE(RET)

The sentence is reinserted as follows:

```
The town of Chilly, Me. is located 40
miles north of Bangor.
The nearest airport is in Bangor,
although a seaplane lands once a week
on nearby Okechibwa Lake.
Major access is by route 193,
which is often snowbound.
[EOB]
```

To return to line editing, use the EXIT command:

```
EXIT[RET]
*
```

If you have a VT52 or VT100 and you want to return to keypad editing, enter the SET KEYPAD command after the asterisk. Type EXIT or QUIT to end the editing session:

```
*SET KEYPAD[RET]
*EXIT[RET]
```

These examples show a small portion of nokeypad editing commands. You may want to experiment with the nokeypad editing examples in Chapter 8.

Chapter 3

Protecting Your Editing Session

This chapter describes the features of EDT that protect your buffer contents: the journal file, the RECOVER qualifier, and error messages.

The Journal File and RECOVER Qualifier

EDT records the edits you make during an editing session in a *journal file*. If your editing session ends abnormally, as may happen in a power failure, you can wait until the system is back in operation and then make use of the journal file. When the operating system prompt appears at the left of the screen, use the *RECOVER* qualifier in the command line to restart EDT and restore your edits.

When you use the RECOVER qualifier, EDT returns to the point in the editing session where you left off. The journal file collects your edits incrementally (by its own units of text storage) rather than continuously. This means that RECOVER will restore all of your work on a file up to the last edit that the journal file recorded. Therefore, some of your text may be lost.

For example, start EDT and enter text as shown. The journal file saves your work whether EDT is in line, nokeypad, or keypad editing; this example uses line editing:

DCL/PDS

```
␣ EDIT/EDT(RET)
-File: LABOR.DAT(RET)
Input file does not exist
[EOB]
*
```

MCR/CCL

```
EDT(RET)
EDT> LABOR.DAT(RET)
Input file does not exist
[EOB]
*
```

```
*INSERT(RET)
```

```
The town has a population of 2,000,(RET)
but its proximity to Albany provides(RET)
an ample labor base of both skilled(RET)
and unskilled persons.(RET)
CTRL Z
```

```
*
```

You can simulate a system failure with the QUIT/SAVE command sequence. The QUIT command deletes the buffer copy of LABOR.DAT, and /SAVE places your edits to the buffer copy into the journal file LABOR.JOU.

NOTE

Typing QUIT/SAVE is not a normal practice; it is used here only to demonstrate the use of the journal file and the RECOVER qualifier.

Enter QUIT/SAVE after the asterisk and press (RET). This sequence ends the editing session:

```
*QUIT/SAVE(RET)
```

Use the RECOVER qualifier as follows:

DCL/PDS

```
EDIT/EDT/RECOVER(RET)
_File: LABOR.DAT(RET)
```

MCR/CCL

```
EDT/RECOVER(RET)
EDT> LABOR.DAT(RET)
```

EDT displays the following when you enter the file name LABOR.DAT:

```
Input file does not exist
INSERT
The town has a population of 2,000,
but its proximity to Albany provides
an ample labor base of both skilled
and unskilled persons.
^Z
QUIT/SAVE

[EOB]
*
```

Display the contents of the buffer to see that what you insert or edit is restored:

```
* TYPE WHOLE(RET)
1      The town has a population of 2,000,
2      but its proximity to Albany provides
3      an ample labor base of both skilled
4      and unskilled persons.
[EOB]
*
```

Note that EDT includes the contents of the journal file as if you had just inserted the edits.

You can continue the editing session after your edits are restored. The journal file retains the input from the previous editing session and continues to store your edits at the end of the file. In this way you can have successive recoveries with one journal file until you end the session with the EXIT or QUIT command.

Editing the Journal File

You can edit the journal file to delete any edits that you did not want inserted. For example, assume you made a mistake in the middle of an editing session, such as typing "D" instead of "T" to display the contents of a buffer:

```
* D WHOLE(RET)
4 lines deleted
[EOB]
*
```

D WHOLE deletes, rather than displays, the contents of LABOR.DAT. You can recover the buffer contents with the following procedure.

Type QUIT/SAVE to keep a copy of your edits without inserting them into the file:

```
* QUIT/SAVE(RET)
```


Start EDT to edit LABOR.JOU. Use the NOJOURNAL qualifier (described in Chapter 4) to prevent EDT from creating a new journal file:

DCL/PDS

```
EDIT/EDT/NOJOURNAL LABOR.JOU(RET)
```

MCR/CCL

```
EDT/NOJOURNAL LABOR.JOU(RET)
```

EDT displays the first line of the journal file:

```
1          INSERT
*  █
```

Display the contents of the journal file as follows:


```
* TYPE WHOLE(RET)
1          INSERT
2          The town has a population of 2,000,
3          but its proximity to Albany provides
4          an ample labor base of both skilled
5          and unskilled persons.
6          ^Z
7          QUIT/SAVE
8          TYPE WHOLE
9          D WHOLE
10
QUIT/SAVE
[EOB]
*  █
```

Delete line 9 to remove D WHOLE from the journal file.


```
* DELETE 9RET
1 line deleted
   10          QUIT/SAVE
* EXITRET
```

Type the command line that starts EDT to include journal file edits to LABOR.DAT:

DCL/PDS

 EDIT/EDT/RECOVER LABOR.DAT^{RET}


MCR/CCL

 EDT/RECOVER LABOR.DAT^{RET}

The following appears on the screen:

```
Input file does not exist
INSERT
The town has a population of 2,000,
but its proximity to Albany provides
an ample labor base of both skilled
and unskilled persons.
^Z
QUIT/SAVE

TYPE WHOLE
   1          The town has a population of 2,000,
   2          but its proximity to Albany provides
   3          an ample labor base of both skilled
   4          and unskilled persons.
[EOB]
QUIT/SAVE

   1          The town has a population of 2,000,
*
```

EDT includes the journal file edits as shown above. Display the buffer contents as follows and type EXIT to end the editing session:

```
*TYPE WHOLE(RET)
  1      The town has a population of 2,000,
  2      but its proximity to Albany provides
  3      an ample labor base of both skilled
  4      and unskilled persons.
[EOB]
*EXIT(RET)
```

Error Messages

EDT contains a set of error messages to warn you when something that it cannot accept has occurred. The error messages point to where the error was detected. Appendix A contains a list of the error messages.

Chapter 4

The Command Line and Startup Command Files

You need to use a *command line* each time you start EDT. The command line gives your operating system the name of the file you want to create or edit. The examples shown so far in this manual use basic command lines. For example, the following command lines let you start EDT to create a file named REPORT.COM:

DCL/PDS

```
EDIT/EDT REPORT.COM(RET)
```

MCR/CCL

```
EDT REPORT.COM(RET)
```

You can also specify qualifiers and startup command files in the command line.

Qualifiers in the command line let you specify which files EDT will work on, and whether or not you want to use a startup command file. (Consult your system documentation for abbreviations of qualifiers.)

You create a *startup command file* if you want EDT to assign certain characteristics to your editing session. For example, a command file could instruct EDT to place text from other files into the main text buffer.


This chapter describes DCL/PDS command lines and qualifiers, and then MCR/CCL command lines and qualifiers, followed by examples.

The last section, which describes startup command files, applies to every system that has EDT.

In this chapter, it is necessary for you to read only the information that applies to your operating system. For example, if your operating system uses only MCR, read the sections describing the MCR and CCL command line and the MCR and CCL command line qualifiers.

The DCL and PDS Command Line

The format of the DCL and PDS command line is:


 EDIT/EDT[/qualifiers] file-spec



where:

EDIT/EDT starts the EDT text editor.

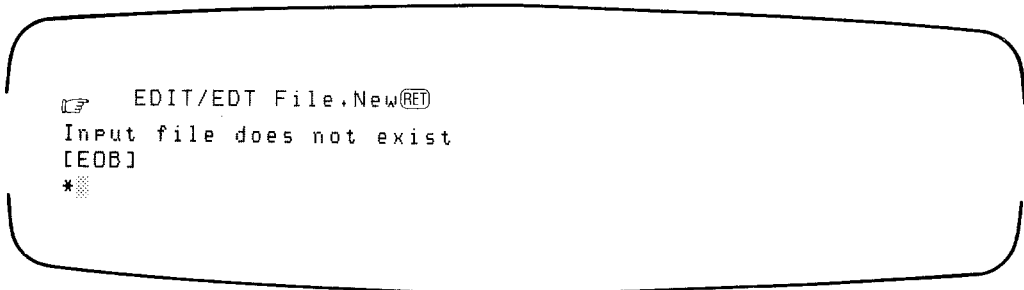
/qualifiers specify command qualifiers.



file-spec specifies the name of the file to be created or edited.
The file is created if it does not already exist.

If you press  after EDIT/EDT without naming a file, you will be prompted for a file name. You must enter an input file name to start EDT:

 EDIT/EDT
_File:

If the input file does not exist, EDT displays the message "Input file does not exist" and starts with an empty buffer. This is how you create a new file with EDT.



```
 EDIT/EDT File.New  
Input file does not exist  
[EOB]  
*
```

DCL and PDS Command Line Qualifiers

You can choose from a set of DCL or PDS command line qualifiers when you start EDT. These qualifiers provide the system with additional information on how to handle EDT text during an editing session.

Table 4-1 lists the DCL/PDS command line qualifiers. You include the slashes when you use the qualifiers in a command line; for example:

```
EDIT/EDT/NOJOURNAL TOWN.DAT(RET)
```

The qualifiers are described in detail following the table.

Table 4-1: DCL and PDS Command Line Qualifiers

| Qualifier | Default |
|--------------------------|---------------------|
| /[NO]OUTPUT[:file-spec] | /OUTPUT:file-spec |
| /[NO]COMMAND[:file-spec] | /COMMAND:EDTINI.EDT |
| /[NO]JOURNAL[:file-spec] | /JOURNAL |
| /[NO]RECOVER | /NORECOVER |
| /[NO]READ_ONLY | /NOREAD_ONLY |

/OUTPUT:file-spec
/NOOUTPUT

The OUTPUT qualifier defines the file specification of the output file created during the editing session. If you do not use OUTPUT:file-spec, the output file has the same name and file type as the input file and a version number one higher than the highest existing version of the file.

/COMMAND[:file-spec]
/NOCOMMAND

The COMMAND qualifier controls whether or not EDT searches for and executes the named file before prompting you for input. If you do not specify COMMAND, or if you enter COMMAND without a file-spec, EDT searches your directory for and executes the contents of file EDTINI.EDT before prompting you for input (see the description of startup command files). If there is no file EDTINI.EDT or you specify /NOCOMMAND, EDT prompts you for input without modifying the default values for the operating parameters.

```
/JOURNAL [:file-spec]  
/NOJOURNAL
```

When you use JOURNAL:file-spec, EDT names the journal file whatever you specify by the file-spec and assigns the extension .JOU. If you specify NOJOURNAL, EDT does not maintain a journal file for that editing session. (See Chapter 3 for more information on journal files.)

```
/RECOVER  
/NORECOVER
```

The RECOVER qualifier controls whether or not the editor will execute the journal file you created in the previous editing session. The rest of the command line must be the same as it was for the command line that started the editing session (see Chapter 3). If you specify NORECOVER, EDT does not attempt to recover with a journal file.

```
/READ_ONLY  
/NOREAD_ONLY
```

The READ_ONLY qualifier specifies that no journal file will be written and no output file produced. With READ_ONLY, if you attempt to exit the editing session without specifying an output file, an error occurs. Use the READ_ONLY qualifier when you do not have write access and want to examine the file without editing it. NOREAD_ONLY, which is the default, specifies that EDT will create an output file and a journal file.

The MCR and CCL Command Line

The format of the MCR/CCL command line is:

```
EDT [[output-file][,journal-file]=]input file [,command file]
```

where:

| | |
|--------------|--|
| EDT | starts the EDT text editor. |
| output-file | is the name you can assign to the updated version of the file. |
| journal-file | is the file containing your edits, which is available if your EDT session ends unexpectedly. |
| input-file | is the name of the file you want to edit. |
| command-file | is the EDT startup file (the default is EDTINI.EDT). |

You must enter an input file name to begin an editing session. EDT will not start until you provide a file name.

```
EDT(RET)  
EDT>
```


When you enter the name of the file you are creating, the system displays a message to show you that you have no other files by that name:

```
EDT File.NewRET  
Input file does not exist  
[EOB]  
*
```

The file you created is FILE.NEW. The output file will have the lowest version number that your system assigns.

Some systems do not use version numbers. If you edit FILE.NEW on a system that does not use version numbers, the updated file is named FILE.NEW and the original input file is renamed FILE.BAK.

MCR and CCL Command Line Qualifiers

You can choose from the set of command qualifiers in Table 4-2 when you start an editing session with either MCR or CCL. These qualifiers, each of which can be abbreviated to the first two characters, can go after any of the file specifications.

You include the slashes when you use the qualifiers in a command line; for example:

```
EDT/NOJOURNAL TOWN.DATRET
```

The hyphen in brackets ([-]) implies the opposite of each qualifier; for example, /-RECOVER means “no RECOVER.”

Table 4-2: MCR and CCL Command Line Qualifiers

| Qualifier | Default |
|--------------|-----------|
| /[-]COMMAND | /COMMAND |
| /[-]JOURNAL | /JOURNAL |
| /[-]RECOVER | /RECOVER |
| /[-]RO | /-RO |
| CCL Only | |
| /[-]VARIABLE | /VARIABLE |
| /[-]STREAM | /STREAM |

The following are descriptions of the MCR and CCL command line qualifiers.

/COMMAND
/-COMMAND

The COMMAND qualifier controls whether or not EDT searches for and executes the named file before prompting you for input. If you do not specify COMMAND, or if you specify COMMAND without a file-spec, EDT searches your directory for and executes the contents of file EDTINI.EDT before prompting you for input (see the description of startup command files). If there is no file EDTINI.EDT or you specify /-COMMAND, EDT prompts you for input without modifying the default values for the operating parameters.

/JOURNAL
/-JOURNAL

The JOURNAL qualifier controls whether or not the journal file is assigned a name other than the input file's name with extension .JOU. If you specify JOURNAL with no file-spec, which is the default, EDT creates a journal file.

If you specify -JOURNAL, EDT does not maintain a journal file for that edit session. (See Chapter 3.)

/RECOVER
/-RECOVER

The RECOVER qualifier controls whether or not the editor will execute the journal file created in the previous editing session. The rest of the command line must be the same as it was for the command line that

started the editing session (see Chapter 3). If you specify **-RECOVER**, EDT does not attempt to recover with a journal file.

/RO
/-RO

The **RO** (Read Only) qualifier specifies that no journal file will be written and no output file produced. With **RO**, if you attempt to exit the editing session without specifying an output file, an error occurs. Use the **RO** qualifier when you do not have write access and want to examine the file without editing it. If you specify **-RO**, EDT creates an output file and a journal file.

The **VARIABLE** and **STREAM** qualifiers are used in **CCL** and do not apply to **MCR**. You cannot specify **VARIABLE** or **STREAM** on the **EXIT**, **WRITE**, or **PRINT** commands. The following are descriptions of the **CCL** only qualifiers.

/VARIABLE
/-VARIABLE

The **VARIABLE** qualifier creates a variable length file. **-VARIABLE** creates a non-variable length, or stream format, file.

/STREAM
/-STREAM

The **STREAM** qualifier creates a stream format file; **-STREAM** creates a variable length file.

Using Command Lines and Qualifiers

The following are examples of command lines and the use of qualifiers.

DCL/PDS

 **EDIT/EDT TEXT.DAT** 

MCR/CCL

 **EDT TEXT.DAT** 

EDT copies the contents of the file **TEXT.DAT** into the main buffer and creates a journal file named **TEXT.JOU**. EDT assigns the name **TEXT.DAT** to the output file.

DCL/PDS

```
EDIT/EDT/OUTPUT:PAYROL.JAN/JOURNAL:REC.ONE PAYROL.FRM
```

MCR/CCL

```
EDT PAYROL.JAN,REC.ONE = PAYROL.FRM
```

EDT copies the contents of the file PAYROL.FRM into the main buffer and opens a journal file named REC.ONE. When you end the editing session with the EXIT command, EDT copies the contents of the main text buffer to the file named PAYROL.JAN. If the editing session ends in other than a normal exit (such as a system failure), EDT retains the journal file REC.ONE.

DCL/PDS

```
EDIT/EDT/OUTPUT:PAYROL.JAN/RECOVER/JOURNAL:REC.ONE PAYROL.FRM
```

MCR/CCL

```
EDT PAYROL.JAN,REC.ONE=PAYROL.FRM/RECOVER
```

EDT copies the contents of the file PAYROL.FRM into the main buffer and then reads the contents of the journal file REC.ONE (the result of a previous editing session) as a command file. At the end of reading the journal file, EDT is in the same state as when the previous editing session ended. The text buffers contain the input files and your input to the terminal. The journal file contains all of your inputs to the terminal. The journal file continues to add to its contents during the recovery editing session. EDT assigns the name PAYROL.JAN to the output file.

DCL/PDS

```
EDIT/EDT/OUTPUT:TEXTA.DAT/COMMAND:FORMC.DAT TEXT.DAT
```

MCR/CCL

```
EDT TEXTA.DAT=TEXT.DAT,FORMC.DAT
```

EDT copies the contents of the file TEXT.DAT into the main buffer and then reads the contents of the command file. After the contents of the command file are executed, EDT prompts you for input. EDT assigns the name TEXTA.DAT to the output file.

Startup Command Files

With EDT, you can create a startup command file to specify characteristics of your editing session, such as the width of a display of text across the screen. You can assign any name to a startup file.

EDT automatically looks for file EDTINI.EDT when you begin an editing session. If you name a startup file something other than EDTINI.EDT, specify the name of the command file in the command line when you start EDT.

NOTE

Extra carriage returns in the command file cause a null command to take effect when the command file is executed. This means that for every blank line in the command file, EDT displays one line of text in the current buffer and moves the cursor down one line.

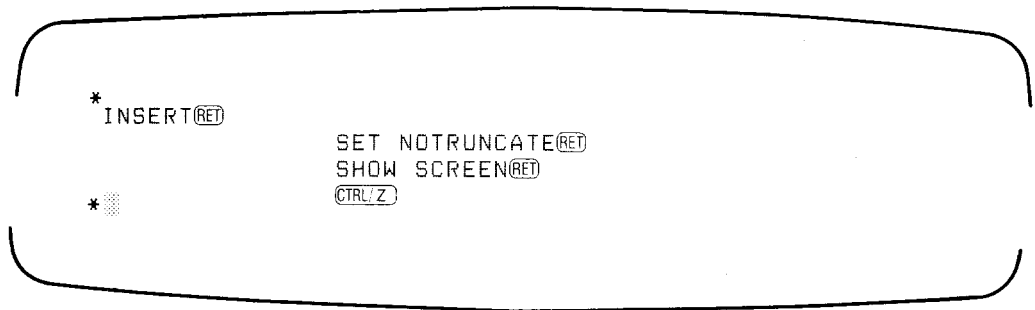
The following is an example of a startup command file, which you create with EDT the same as you would any other file.

TABLEA.COM

```
* INSERT␣  
  
SET ENTITY PAGE '.PAGE'␣  
SET WRAP 65␣  
INCLUDE HEAD1.COM =HEAD1␣  
INCLUDE HEAD5.COM =HEAD5␣  
INCLUDE DATA.FIL =MAIN␣  
␣  
* ␣
```

When you specify the TABLEA.COM file as the command file in the command line, EDT executes the commands in the file and prompts you for input. With this command file, you have set the user-definable entity page to .PAGE and you have set the right margin for a word wrap of 65 characters. EDT loads the files HEAD1.COM and HEAD5.COM into text buffers named HEAD1 and HEAD5, respectively. EDT loads the file DATA.FIL into the main buffer just ahead of the file loaded into the main buffer by the command line.

EDTINI.EDT



The EDTINI.EDT file is the default command file, so you do not have to name it in the command line. If you have this EDTINI.EDT file when you start EDT, SHOW SCREEN causes EDT to display the maximum number of characters you can insert on a line. SET NOTRUNCATE means that if you type more than the maximum number of characters before you press ␣, the remaining characters will be displayed on the next line.

Chapter 5

Keypad Editing

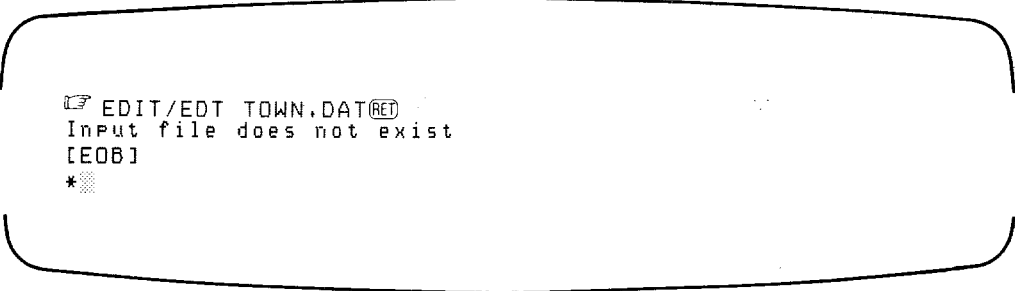
This chapter describes keypad editing, which is available on VT52 and VT100 terminals. Because the keypad keys are not labeled with what they do, this chapter discusses keypad functions rather than keys.

A key is found on the keyboard or keypad and is labeled with characters such as “,” or “ENTER”. By contrast, a function is the operation performed when you press that key. For example, the SUBSTITUTE and ENTER functions are associated with the ENTER key on the keypad.

Starting an Editing Session

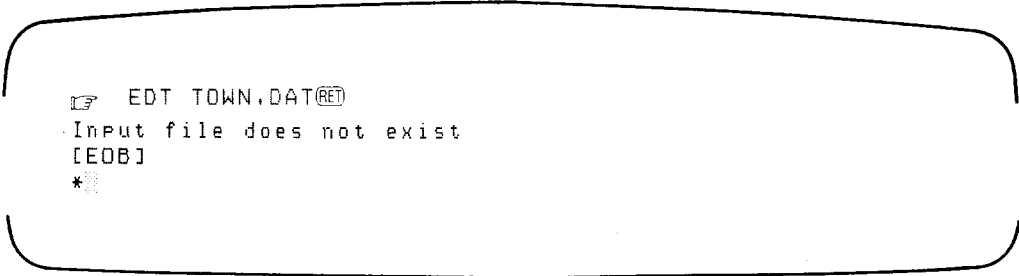
Start EDT with the appropriate command line:

DCL/PDS



```
EDIT/EDT TOWN.DAT
Input file does not exist
[EOB]
*
```

MCR/CCL



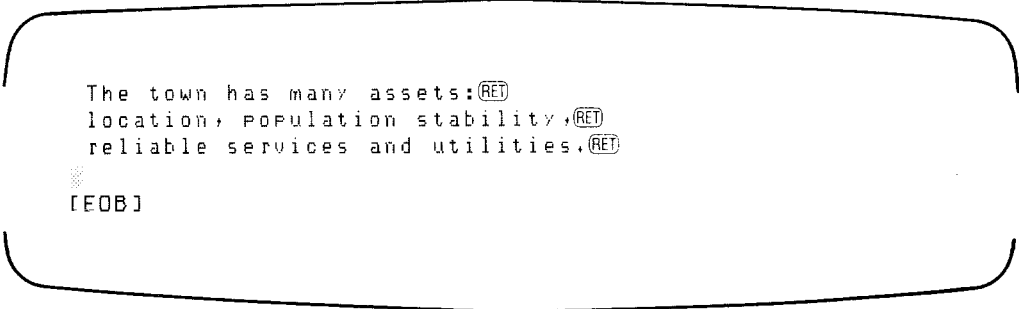
```
EDT TOWN.DAT␣  
Input file does not exist  
[EOB]  
*
```

You can begin keypad editing as soon as you type the **CHANGE** command after the asterisk:

```
* CHANGE␣
```

If you are editing an existing file, a copy of the file's contents appears on the screen. If you are creating a new file, the screen clears for your input. All entries that you make on the keyboard are entered as text before the cursor.

In this example you are creating the file **TOWN.DAT**. The screen is blank, since there is no text in the new file. The cursor appears in the upper left of the screen. The message **[EOB]** shows that you are at the end of the buffer in the new file. What you type appears before the cursor. Enter the following text:



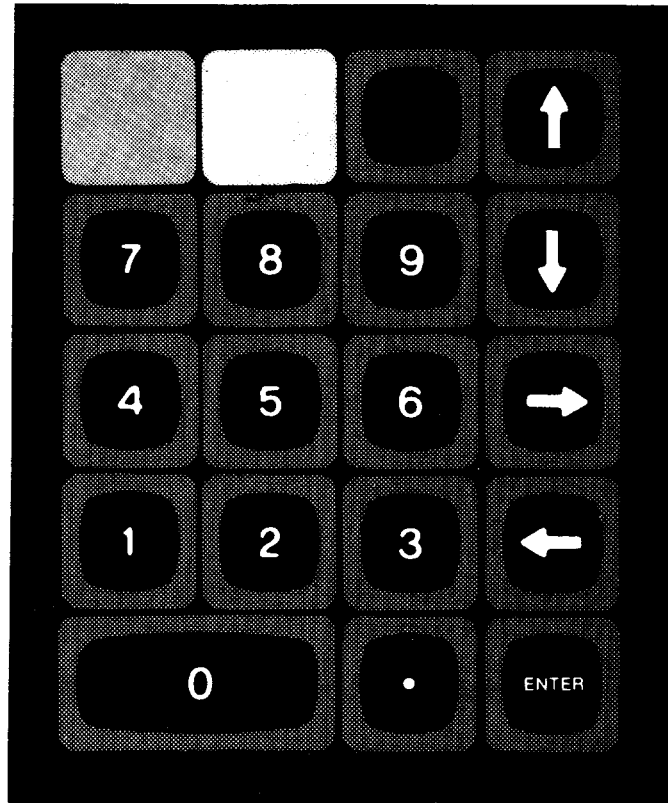
```
The town has many assets:␣  
location, population stability,␣  
reliable services and utilities.␣  
[EOB]
```

Using the Keypad

The keypad is the set of keys at the right of the keyboard. Figure 5-1 shows the VT52 keypad and its corresponding functions. Figure 5-2 shows the keypad and functions of the VT100.

Figure 5-1: The VT52 Keypad and Its Functions

A. The Keypad

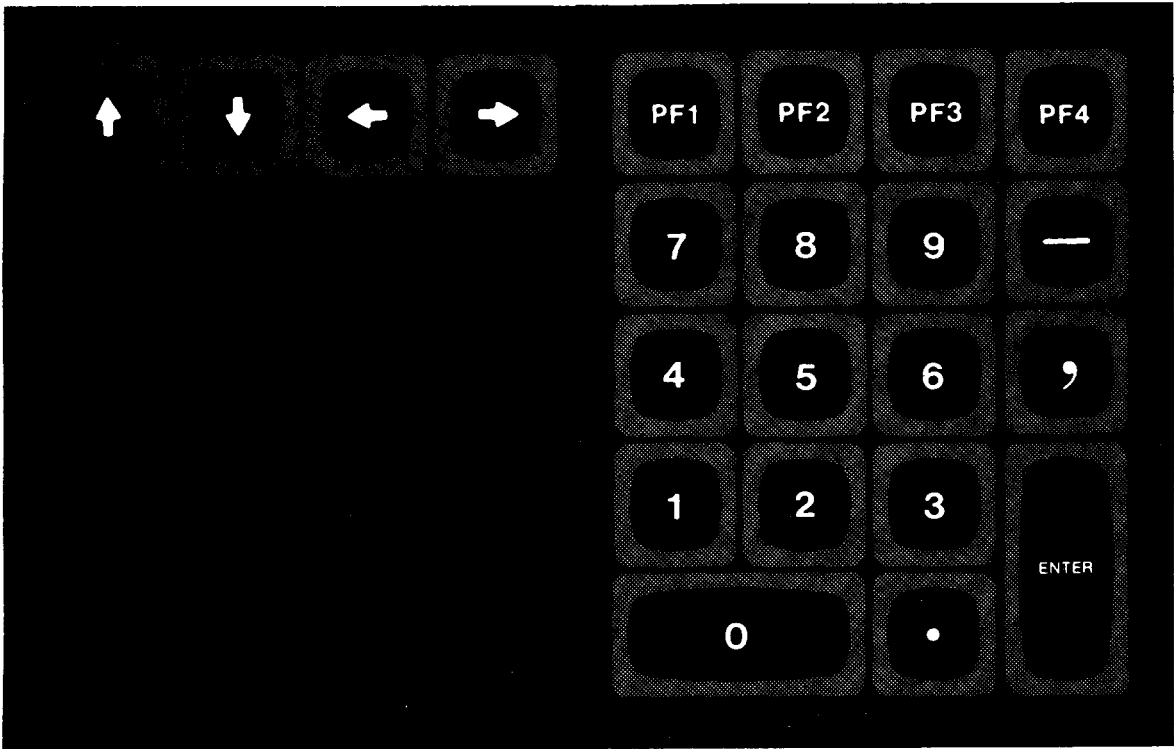


B. Keypad Functions

| | | | |
|-------------------|----------------|-----------------|------------------|
| GOLD | HELP | DEL L UND L | UP REPLACE |
| PAGE COMMAND | FNDNXT FIND | DEL W UND W | DOWN SECT |
| ADVANCE BOTTOM | BACKUP TOP | DEL C UND C | RIGHT SPECINS |
| WORD CHNGCASE | EOL DEL EOL | CUT PASTE | LEFT APPEND |
| LINE OPEN LINE | | SELECT RESET | ENTER SUBS |

Figure 5-2: The VT100 Keypad and Its Functions

A. The Keypad and the Arrow Keys



B. Keypad Functions

| | | | |
|----|------|------|-------|
| UP | DOWN | LEFT | RIGHT |
|----|------|------|-------|

| | | | |
|-------------------|----------------|-------------------|----------------|
| GOLD | HELP | FNDNXT FIND | DEL L UND L |
| PAGE COMMAND | SECT FILL | APPEND REPLACE | DEL W UND W |
| ADVANCE BOTTOM | BACKUP TOP | CUT PASTE | DEL C UND C |
| WORD CHNGCASE | EOL DEL EOL | CHAR SPECINS | ENTER SUBS |
| LINE OPEN LINE | | SELECT RESET | |

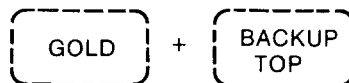
Keypad keys are labeled with characters such as ".", "ENTER", or "5". Functions are assigned to each of the keypad keys. For example, you use the "5" key for BACKUP and TOP functions

Figures 5-1 and 5-2 show the four arrow keys that are available for screen editing. These arrow keys let you move the cursor about in the text. In addition, the LINE FEED and BACK SPACE keys have special functions in keypad mode.

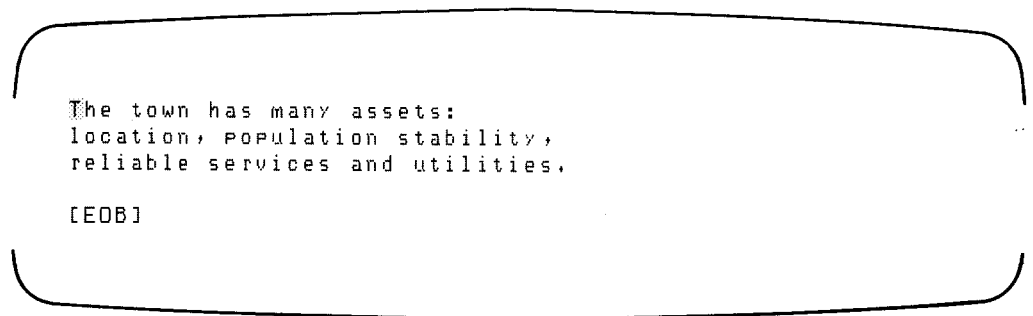
You can also use CONTROL functions by pressing the CONTROL key (CTRL) and one of several alphabetic character keys. For example, if you want to use the CTRL/W, press the CONTROL key while holding down the "W" key on the keyboard.

Notice that most of the keys in Figures 5-1 and 5-2 have two functions. You can use either of the two functions, depending on whether or not you press the GOLD key first. (The GOLD key is at the upper left of both VT52 and VT100 keypads.) To use the standard function (shown as the upper of the two functions on a key), press the key. To use the alternate (lower) function, press GOLD first and then the key.

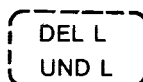
For example, press GOLD and then the TOP function key (the keypad key labeled "5"). TOP is the alternate function of the key:



Note that the cursor has moved to the character at the upper left of the screen:



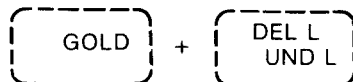
Press the key on your terminal's keypad that corresponds to the DEL L function, and watch what happens to your text:



location, population stability,
reliable services and utilities.

[EOB]

Now press GOLD and the UND L function key to restore the line:

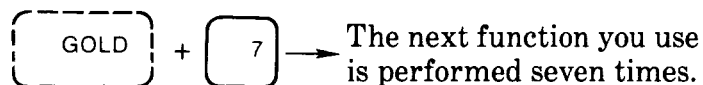


The town has many assets:
location, population stability,
reliable services and utilities.

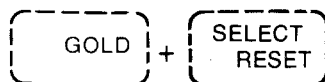
[EOB]

Throughout this chapter, a statement such as "Use the UND L function" assumes that you press GOLD and then the UND L function key.

In this chapter, the keys in the diagrams are enclosed by solid lines. Functions are shown with dashed lines. The result of a keypad operation is shown with an arrow. For example, if you are told to "press the GOLD function key on the keypad and then the 7 key on the keyboard," the diagram looks like this:



(The GOLD integer function is explained in the next section.) To continue with examples in this chapter, press GOLD and RESET (the keypad key labeled ".") to cancel the GOLD and "7" key sequence:



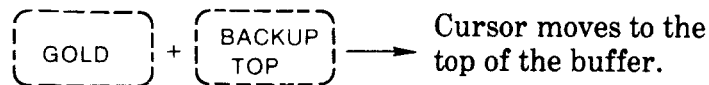
Essential Functions

These functions are either required for or basic to keypad operations.

GOLD

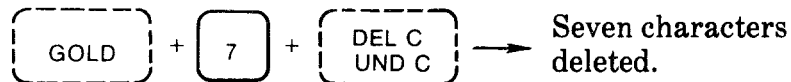
The GOLD function causes EDT to perform the alternate function on any of the keypad keys. To use GOLD, press the GOLD function key and then a keypad key.

The following example shows how GOLD is used with the TOP function:



As shown above, when you press the GOLD key and then another key, the alternate function (TOP in this example) is performed. GOLD remains in effect until you press one of the other keypad keys.

GOLD also lets you execute a function a given number of times. First, press GOLD and enter a number from the keyboard. This number appears at the bottom left of the screen and remains until you complete the operation. Then press a keypad key for the function you want performed:



The first line of the buffer now reads:

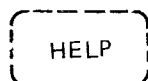
```
n has many assets:
```

You can reinsert these characters by typing them:

```
The town has many assets:
```

HELP

Pressing the HELP function key causes a diagram of keypad functions and CONTROL key descriptions to appear on your screen:



At the bottom of the HELP diagram are descriptions on how to get further help or to return to your editing session. At this point you can:

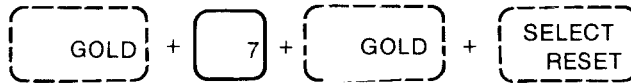
- Get help on another keypad key by pressing the key
- Return to the keypad diagram by pressing **RET**
- Exit from HELP by typing a space

When you press one of the keypad keys, EDT displays a description of the standard and alternate functions of that key. When you exit from HELP, the cursor returns to where it was in your editing session before you used HELP.

RESET

Use RESET to cancel the effects of the GOLD and SELECT functions or any key sequence that is partially entered. To use RESET, press GOLD and then the RESET function key.

For example, suppose you enter a sequence of functions and then decide against using it. The following example shows how to cancel the sequence (note that "7" is a key on the keyboard):



The GOLD and RESET key sequence cancels the repeat operation. You can use the RESET function to cancel any operation that has not yet been executed. RESET will not undo an operation once it is performed.

Refreshing the Screen (CTRL/W)

CTRL/W refreshes the screen display. The screen becomes blank and then the characters in the buffer reappear:

CTRL/W

Refreshing the screen deletes extraneous characters, such as from system messages, that might appear on the screen during your editing session. The cursor remains in the same location during a refresh operation.

Returning to Line Editing (CTRL/Z)

CTRL/Z returns you to the line editing asterisk (*) prompt:

CTRL/Z

To resume keypad editing, type CHANGE after the asterisk and press (RET) :

*CHANGE(RET)

Ending an Editing Session

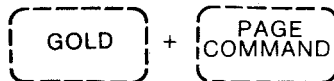
When you want to stop keypad editing, you enter commands that either save your work from the editing session or delete what you have done.

The edits that you make during an EDT editing session are stored in a text buffer. When you “save” or “delete” edits, you are really saving or deleting the contents of the text buffer. Therefore, when you create a new file, you decide whether or not to store the text buffer copy as the new file. When you work on a copy of an existing file, you decide whether to incorporate your edits into the file or to delete your work from the editing session.

Saving Your Edits

You can save the edits you make during a keypad editing session two ways:

1. Press GOLD and then the COMMAND function key. Type EXIT after the prompt and press the ENTER function key:



COMMAND: EXIT

A diagram of a rectangular key with a dashed border containing the words "ENTER" and "SUBS" stacked vertically.

2. Press (CTRL/Z) to return to line editing. Type EXIT after the asterisk and then press (RET) .

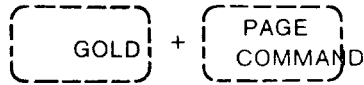
Whether you type EXIT after the COMMAND prompt or in line editing, you save the file with your edits and exit EDT.

Deleting your Edits

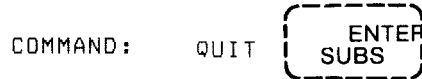
You can delete all the edits you make during an editing session with the QUIT command. QUIT works both for new and existing files. For example, assume you start EDT to edit an existing file. If you make mistakes or are dissatisfied with your edits, you may find it easier to use the QUIT command and then start a new editing session.

There are two ways to delete your edits with QUIT. (If you tried EXIT in the previous section and now want to try QUIT, restart EDT to edit any file.)

1. Press GOLD and then the COMMAND function key:



Type QUIT after the prompt and press the ENTER function key:



2. Return to line editing by pressing **CTRL/Z**. Then type QUIT after the asterisk.

Either method deletes all your edits and ends your session with EDT.

Inserting Text

You can enter text into the file as soon as you start keypad editing.

Inserting Characters

Whatever you type on the keyboard during keypad editing is inserted directly before the cursor as text. Start EDT, then enter change mode by typing CHANGE after the asterisk prompt and pressing **RET**:

DCL/PDS

A screenshot of a terminal window with a rounded rectangular border. The text inside is as follows:
EDIT/EDT CASTLE.DAT^{RET}
Input file does not exist
[EOB]
* CHANGE^{RET}

MCR/CCL

```
EDT CASTLE.DATRET  
Input file does not exist  
[EOB]  
* CHANGERET
```

Then type in text at the keyboard:

```
Castle Island, a small fishing communityRET  
on the coast of Maine,RET  
is a popular tourist attractionRET  
because of its quaint restaurants andRET  
its outdoor fish markets.RET  
[EOB]
```

To try the following examples, press the UP and WORD function keys as shown. (These functions will be explained later.)

↑ (3 times) + WORD
CHNGCASE (3 times)

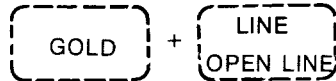
The cursor is now in the middle of the third line:

```
is a popular tourist attraction
```

Using OPEN LINE

You can use OPEN LINE to insert a carriage return after the cursor. The cursor position does not change. If the cursor is in the middle of a

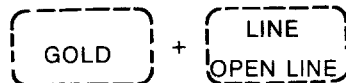
line, EDT moves any text at the right of the cursor down to the beginning of the next line:



```
Castle Island, a small fishing community
on the coast of Maine,
is a popular
tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

[EOB]
```

If the cursor is at the beginning or end of a line, the effect of this function is to insert a blank line:



```
Castle Island, a small fishing community
on the coast of Maine,
is a popular

tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

[EOB]
```

Moving the Cursor

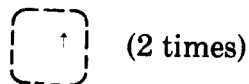
The functions in this section let you move the cursor and set the direction of cursor movement during your editing session.

The Arrow Keys

You can move the cursor up, down, to the left, and to the right with the four arrow keys. These keys are found on the VT52 keypad and on the VT100 keyboard.



The UP (up arrow) function key moves the cursor to the character in the line above. If the line above does not extend to the present cursor position, EDT places the cursor at the end of the line. The cursor moves to the original character space when successive lines extend that far. For example:



```
Castle Island, a small fishing community  
on the coast of Maine,  
is a popular  
  
tourist attraction  
because of its quaint restaurants and  
its outdoor fish markets.  
  
[EOB]
```



The DOWN (down arrow) function key moves the cursor to the character below. If the line does not extend to the present cursor position, EDT places the cursor at the end of the line. The cursor moves to the original character space when successive lines extend that far.



Castle Island, a small fishing community
on the coast of Maine,
is a popular

tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

[EOB]

LEFT



The **LEFT** (left arrow) function key moves the cursor to the preceding character. When the cursor is on the left margin, EDT moves the cursor after the rightmost character on the line above.



(3 times)

Castle Island, a small fishing community
on the coast of Maine,
is a popular

tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

[EOB]

RIGHT



The **RIGHT** (right arrow) function key moves the cursor to the next character. When the cursor is after the rightmost character of the line, EDT moves the cursor to the first character on the next line.



(3 times)

Castle Island, a small fishing community
on the coast of Maine,
is a popular

tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

[EOB]

Setting the Direction of Cursor Movement

ADVANCE and BACKUP change the direction in which the cursor moves. The following functions are affected by ADVANCE and BACKUP:

| | | | | |
|------|------|------|---------|------------|
| CHAR | WORD | FIND | FNDNXT | CHNGCASE |
| LINE | EOL | PAGE | SECTION | SUBSTITUTE |
| | | | | TAB ADJUST |

ADVANCE

ADVANCE sets the cursor direction forward. The cursor direction is toward the end of the file, that is, to the right and down. To use ADVANCE, press the key:

ADVANCE
BOTTOM

ADVANCE is the default direction of cursor movement. This default remains in effect until you press BACKUP.

BACKUP

BACKUP sets the cursor direction backward. The direction of cursor movement is toward the start of the file, that is, to the left and up. To use BACKUP, press the key.

BACKUP
TOP

When you select BACKUP, it remains in effect until you press the ADVANCE function key:



Movement by Entity

You can move the cursor by entities of text. An entity is a character string that EDT recognizes as a unit, such as a word or paragraph. The direction of cursor movement by entity depends on whether EDT is set to ADVANCE or BACKUP.

CHAR (Character)

CHAR moves the cursor one character to the right or left. When necessary, the cursor moves up or down a line to move to the next character. The CHAR function is available with the VT100 only.

For example:



(3 times)

```
Castle Island, a small fishing community  
on the coast of Maine,  
is a popular  
  
tourist attraction  
because of its quaint restaurants and  
its outdoor fish markets.  
  
[EOB]
```

CHAR differs from the left and right arrow keys in that CHAR is affected by the ADVANCE and BACKUP functions, while the arrow keys are not.

WORD

A word in EDT is one or more characters that are preceded and followed by spaces. WORD moves the cursor one word forward or one word back. For example:

WORD
CHNGCASE (3 times)

Castle Island, a small fishing community
on the coast of Maine,
is a popular

tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

[EOB]

When the cursor is at the end of a line and the direction for the move is beyond the end of the line, the cursor moves to the next line:

WORD
CHNGCASE (3 times)

Castle Island, a small fishing community
on the coast of Maine,
is a popular

tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

[EOB]

EOL (End Of Line)

EOL moves the cursor forward to the end of the current line or backward to the end of the previous line. The cursor movement includes any blank spaces before the last line terminator. For example:

EOL
DEL EOL (2 times)

Castle Island, a small fishing community
on the coast of Maine,
is a popular

tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

[EOB]

LINE

LINE moves the cursor down to the start of the next line or back to the start of the current line (in other words, to the left hand margin). For example:

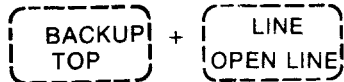
LINE
OPEN LINE

Castle Island, a small fishing community
on the coast of Maine,
is a popular

tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

[EOB]

Press the BACKUP function key to reverse the direction of cursor movement:



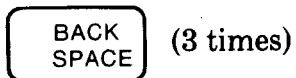
Castle Island, a small fishing community
on the coast of Maine,
is a popular

tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

[EOB]

BACK SPACE

BACK SPACE places the cursor at the beginning of the line, that is, at the leftmost character position. For example:



Castle Island, a small fishing community
on the coast of Maine,
is a popular

tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

[EOB]

Movement Throughout the Buffer

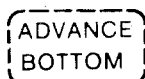
You can move the cursor to the top or bottom of the text buffer, or by sections and pages.

PAGE

PAGE moves the cursor to the top of the next page or previous page. To use PAGE, press the key:



You can define the page delimiter with the SET command (see Chapter 9). The default delimiter for a page is form feed, an ASCII character that determines the start of each line printer page. (Refer to Appendix B for a list of ASCII character codes.) The direction of cursor movement depends on whether EDT is set to ADVANCE or BACKUP. For the next example, press the ADVANCE function key:



SECTION

SECTION moves the cursor one 16-line section. To use SECTION, press the key:



If there are fewer than 16 lines for the cursor to move, EDT moves the cursor as many lines as it can and then displays one of the following messages:

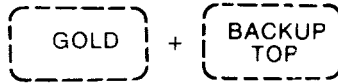
Backup past top of buffer

Advance past bottom of buffer

The direction of cursor movement depends on whether EDT is set to ADVANCE or BACKUP.

TOP

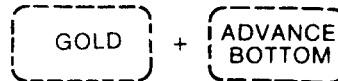
TOP positions the cursor at the top of the text buffer. To use TOP, press GOLD and the TOP function key:



EDT displays text for as many lines as you set with the SET LINES command (see Chapter 9). If you do not set the number of lines to be displayed on the screen, the default of 22 lines appears.

BOTTOM

BOTTOM positions the cursor at the bottom of the text buffer, directly preceding the [EOB] symbol. To use BOTTOM, press GOLD and the BOTTOM function key:



EDT displays text for as many lines as you set with the SET LINES command (see Chapter 9). If you do not set the number of lines to be displayed on the screen, the default of 22 lines appears.

Locating Text

The functions in this section let you locate strings of characters.

FIND

Press GOLD and then the FIND function key to use this function. FIND prompts with "Search for:" on the lower left of your screen. You enter the desired string through the keyboard and end the string with one of the following:

- ADVANCE or BACKUP to set the direction of search
- ENTER to use the direction currently specified

For example:

```
Castle Island, a small fishing community
on the coast of Maine,
is a popular

tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

[EOB]
```

+ Search for: *

After the prompt, enter the text you want to find and set the direction of the search with ADVANCE or BACKUP:

Search for: is

The cursor moves to the first character in the search string:

```
Castle Island, a small fishing community
on the coast of Maine,
is a popular

tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

[EOB]
```

When you use ADVANCE or BACKUP to set the direction of a search, you also determine the direction of cursor movement for other operations, such as EOL or SECTION.

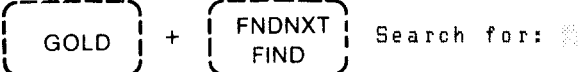
The string you enter in response to the FIND prompt is stored in the search string buffer. This buffer is also used for the FNDNXT, CUT, SUBSTITUTE, CHNGCASE, and REPLACE functions.

FNDNXT (Find Next)

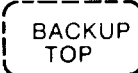
FNDNXT searches for the string that is stored in the search buffer. For example, if the last search string you used in a FIND operation was "is", pressing FNDNXT would cause a search for the next occurrence of "is".

You can search forward or backward; the direction depends on whether EDT is set to ADVANCE or BACKUP. EDT stores the search string in a buffer until you enter a new search string or exit EDT.

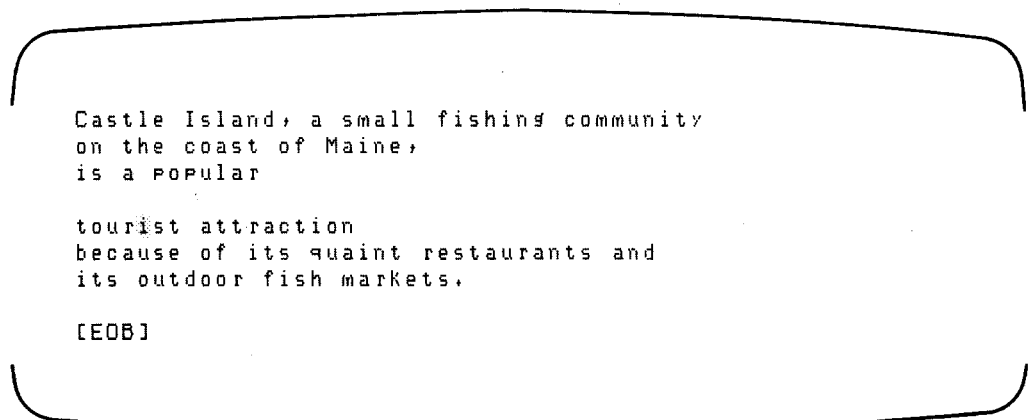
For example, you can use FIND and FNDNXT to search for occurrences of the word "is" in the buffer. When you press GOLD and the FIND function key, you are prompted for a search string:

A diagram showing two dashed boxes. The first box contains the word "GOLD". To its right is a plus sign "+". The second box contains the words "FNDNXT" and "FIND" stacked vertically. To the right of the second box is the text "Search for:" followed by a small cursor icon.

Type "is" and press BACKUP:

Search for: is A diagram showing the text "Search for: is" followed by a dashed box containing the words "BACKUP" and "STOP" stacked vertically.

The cursor moves to the first occurrence of the search string:

A large rounded rectangle representing a buffer. Inside, the text is as follows:
Castle Island, a small fishing community
on the coast of Maine,
is a popular

tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

[EOB]
The cursor is positioned at the start of the word "is" on the third line.

Pressing the FNDNXT function key moves the cursor to the next occurrence:

FNDNXT
FIND

Castle Island, a small fishing community
on the coast of Maine,
is a popular

tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

[EOB]

You can repeat the FNDNXT function:

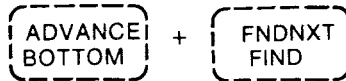
FNDNXT
FIND

Castle Island, a small fishing community
on the coast of Maine,
is a popular

tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

[EOB]

If you want to search forward, press the ADVANCE function key and then press FNDNXT:



Castle Island, a small fishing community
on the coast of Maine,
is a popular

tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

[EOB]

Deleting and Reinserting Text

The functions in this section let you delete text from a buffer and reinsert deleted text. Each of the functions that delete characters, words, or lines stores the deleted text in buffers for possible reinsertion. The functions are also useful for moving the deletions about in the buffer.

Deleting and Reinserting by Character

This section describes how to delete and undelete text by individual characters.

DELETE

The DELETE key, which is located on the keyboard, deletes the character to the immediate left of the cursor and stores it in the character buffer. You can later reinsert the character in this buffer, until another deleted character takes its place in the buffer.

When the cursor is at the leftmost character position on a line, EDT deletes the line terminator to the left and moves the text on the line to the right of the text in the line above.

In the following example, you delete the carriage return preceding the third line:

DEL

```
Castle Island, a small fishing community
on the coast of Maine, is a popular
tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

[EOB]
```

Reinsert what you deleted by pressing **RET**:

```
Castle Island, a small fishing community
on the coast of Maine,
is a popular
tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

[EOB]
```

DEL C (Delete Character)

DEL C lets you delete the character that the cursor is on and stores it in the character buffer. For example:

```
is a popular
```

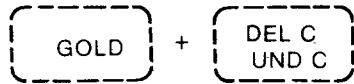
DEL C
UND C

```
s a popular
```


UND C (Undelete Character)

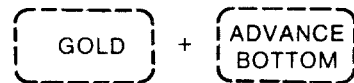
UND C reinserts the character that was deleted by the last DEL C or DELETE command. You are therefore able to replace a character that you accidentally deleted. For example:

s a POPular



is a POPular

DEL C and UND C are useful for correcting typographical errors. For example, if you mistyped "Castle", you could use DEL C and UND C to correct it. Move the cursor to the bottom of the buffer with GOLD and the BOTTOM function key, and type the sentence as shown to follow the example:



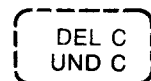
Castel Island, a small fishing community^{RET}
[EOB]

Move the cursor to the "e" in "Castel":



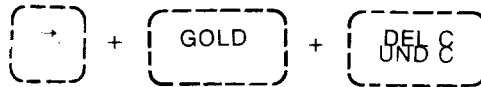
Castel Island, a small fishing community

Now store the letter "e" in the character buffer by pressing the DEL C function key:



Castl Island, a small fishing community

Move the cursor one space to the right and then press GOLD and UND C:

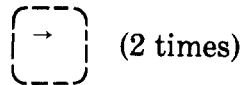


Castle Island, a small fishing community

Deleting and Reinserting by Word

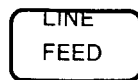
LINE FEED

LINE FEED deletes the characters from the cursor back to the beginning of the word. EDT does not delete the character that the cursor is on. When the cursor is at the first letter in a word, EDT deletes the preceding word and the spaces in between. Press the RIGHT arrow function key twice for the next example:



Castle Island, a small fishing community

Use the LINE FEED function as shown:

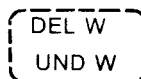


Island, a small fishing community

DEL W (Delete Word)

DEL W deletes the text up to the first character of the next word. Each DEL W operation clears the contents of the word buffer and inserts a new entry.

Use DEL W as shown:

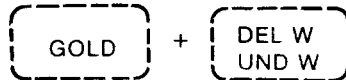


a small fishing community

The word buffer now contains "Island," which replaces the text that was stored during the previous DEL W operation.

UND W (Undelete Word)

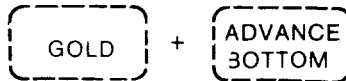
UND W inserts the contents of the word buffer in front of the cursor in the current buffer. The word buffer contains the text deleted by the last DEL W (keypad function) or LINE FEED (keyboard command). You are therefore able to replace a word that you accidentally deleted.



Island, a small fishing community

DEL W and UND W are especially useful when words are misplaced. You can delete a word, move the cursor to another location, and then undelete the word.

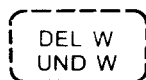
For this example, move the cursor to the end of the buffer:



Type the following sentence and move the cursor to the "A" in "Acton":

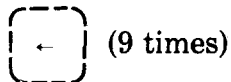
The first three sites are Boxboro, Acton, and Concord.

1. Press the DEL W function key to store "Acton," in the word buffer:



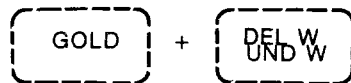
The first three sites are Boxboro, and Concord.

2. Move the cursor to the place where you want to reinsert the deleted word:



The first three sites are Boxboro, and Concord.

3. Press GOLD and then the UND W function key:



The first three sites are Acton, Boxboro, and Concord.

Deleting and Reinserting by Line

CTRL/U

CTRL/U deletes the text from the cursor position to the beginning of the line. When the cursor is at the leftmost character position on a line, EDT deletes the line above. EDT loads the deleted text into the line buffer.

The first three sites are Acton, Boxboro, and Concord.

CTRL/U

Acton, Boxboro, and Concord.

DEL EOL (Delete to End of Line)

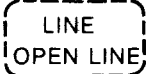

DEL EOL deletes all the characters to the right of the cursor, up to the end of the line, including the character on which the cursor is positioned. If you are at the end of a line, DEL EOL deletes the next line.

Type the following text for this example:



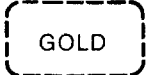
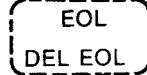
Acton, Boxboro, and Concord.
These three neighboring towns (RET)
are in a prestigious area of (RET)
the state. (RET)
[EOB]

Move the cursor to the beginning of the second line:

 +  (3 times)

```
Acton, Buxboro, and Concord.  
These three neighboring towns  
are in a Prestigious area of  
the state.  
[EOB]
```

If you press GOLD and the DEL EOL function key, the line is deleted:

 + 

```
Acton, Buxboro, and Concord.  
are in a Prestigious area of  
the state.  
[EOB]
```

Retype the line to follow the next example:

```
Acton, Buxboro, and Concord.  
These three neighboring towns  
are in a Prestigious area of  
the state.  
[EOB]
```

DEL L (Delete through End of Line)

DEL L deletes the text from the cursor position through the next line terminator. The first character of the following line moves up to the cursor.

DEL L
UND L

```
Acton, Boxboro, and Concord.  
are in a prestigious area of  
the state.  
[EOB]
```

UND L (Undelete Line)

UND L reinserts the text deleted by the last CTRL/U, DEL EOL, or DEL L command. You can therefore replace a line that you accidentally deleted.

The following example shows how UND L restores the contents of the buffer after the DEL L function:

GOLD + DEL L
UND L

```
Acton, Boxboro, and Concord.  
These three neighboring towns  
are in a prestigious area of  
the state.  
[EOB]
```

Selecting and Moving Text

This section describes how to use a select range for moving text about in a text buffer. The example shown at the end of the SELECT, CUT, and PASTE descriptions shows how to use the three functions.

SELECT

SELECT lets you mark one end of a string of text that you want to delete or move. You mark one end of the string by pressing the SELECT function key. Then you move the cursor, either backwards or forwards, to the other end of the string and press the CUT, APPEND, or REPLACE function key.

On a VT100 you can see what characters are in the range you select because a VT100 select range appears in reverse video. VT52 terminals also allow select ranges, although on a VT52 the select range is not in reverse video.

CUT

CUT deletes the selected string from the file and stores it in the paste buffer. This function is especially convenient when you want to delete or move large sections of text. The selected string is all the text between the cursor location when you pressed SELECT and the position to which you moved the cursor.

PASTE

PASTE inserts the contents of the paste buffer in front of the cursor position.

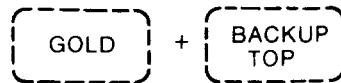
Using SELECT, CUT, and PASTE

Delete the two blank lines in the "Castle Island" text, and move the cursor to the top of the buffer. Then enter text as shown:

GOLD + BACKUP
TOP + GOLD + LINE
OPEN LINE

```
The Eastern Division may establish a branch(RET)
office in a historic New England village.(RET)
Castle Island, a small fishing community
on the coast of Maine,
is a popular tourist attraction
because of its quaint restaurants and
its outdoor fish markets.
```

Move the cursor to the start of the text to follow the example:



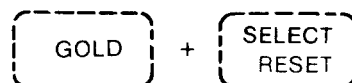
This example shows how to select a range of text and how to clear the range if you make a mistake. Next, it shows the use of CUT to delete the text in the select range and how to reinsert the text with PASTE.

1. **SELECT** — Press the **SELECT** function key to mark the start of the select range, then move the cursor down two lines:



The Eastern Division may establish a branch
office in a historic New England village.
Castle Island, a small fishing community
on the coast of Maine,
is a popular tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

2. **RESET** — Assume you made a mistake and cancel the select range by pressing **GOLD** and **RESET**:



The Eastern Division may establish a branch
office in a historic New England village.
Castle Island, a small fishing community
on the coast of Maine,
is a popular tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

3. CUT — You need a select range to use CUT. Otherwise, when you press the CUT function key, EDT displays the following message:

No select range active

Press the SELECT function key and move the cursor to the end of the sentence:

 +  (5 times)

The Eastern Division may establish a branch office in a historic New England village. Castle Island, a small fishing community on the coast of Maine, is a popular tourist attraction because of its quaint restaurants and its outdoor fish markets.

When you press the CUT function key, the text is moved to the paste buffer:



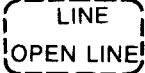
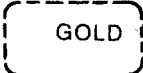
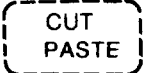
The Eastern Division may establish a branch office in a historic New England village.

4. PASTE — Move the cursor to the location where you want the text inserted, and then press GOLD and the PASTE function key:

 (2 times) +  + 



Castle Island, a small fishing community
on the coast of Maine,
is a popular tourist attraction
because of its quaint restaurants and
its outdoor fish markets.
The Eastern Division may establish a branch
office in a historic New England village.

You can reinsert the text as many times as you like, until you replace the contents of the paste buffer in another operation (such as CUT or APPEND):

 (2 times) +  + 

Castle Island, a small fishing community
on the coast of Maine,
is a popular tourist attraction
because of its quaint restaurants and
its outdoor fish markets.
The Eastern Division may establish a branch
office in a historic New England village.
Castle Island, a small fishing community
on the coast of Maine,
is a popular tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

5. Reverse CUT — Press the SELECT function key and move the cursor up five lines:

 +  (5 times)

Castle Island, a small fishing community
on the coast of Maine,
is a popular tourist attraction
because of its quaint restaurants and
its outdoor fish markets.
The Eastern Division may establish a branch
office in a historic New England village.
Castle Island, a small fishing community
on the coast of Maine,
is a popular tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

When you press CUT, the text in the select range disappears from the screen. The text from this CUT operation replaces the contents of the paste buffer:

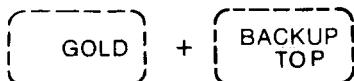


Castle Island, a small fishing community
on the coast of Maine,
is a popular tourist attraction
because of its quaint restaurants and
its outdoor fish markets.
The Eastern Division may establish a branch
office in a historic New England village.

APPEND

APPEND deletes the selected string from the current buffer and stores it at the end of the paste buffer. The selected string is the text between the SELECT entry and the cursor position at the time when you press APPEND.

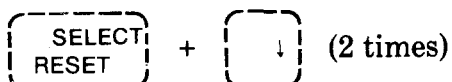
For example, assume that you want to append the fourth line to the first two lines in the following example and move all three lines to the end of the text buffer. Move the cursor to the top of the buffer by pressing the GOLD and TOP function keys:



A diagram of a text buffer represented by a rounded rectangle. Inside, there are four lines of text:
Castle Island, a small fishing community
on the coast of Maine,
is a popular tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

The procedure to append is as follows:

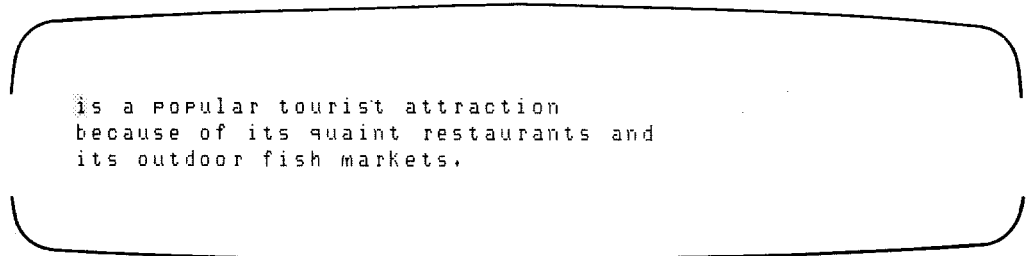
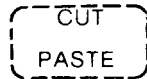
1. Mark the first two lines with a select range:



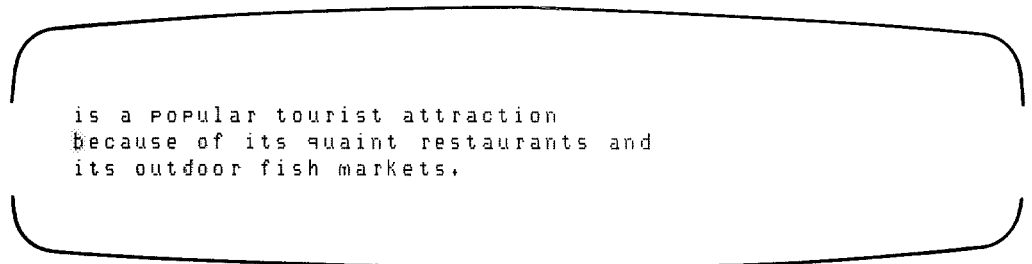
A diagram of a text buffer represented by a rounded rectangle. Inside, there are four lines of text:
Castle Island, a small fishing community
on the coast of Maine,
is a popular tourist attraction
because of its quaint restaurants and
its outdoor fish markets.

The first two lines of text are highlighted with a gray background.

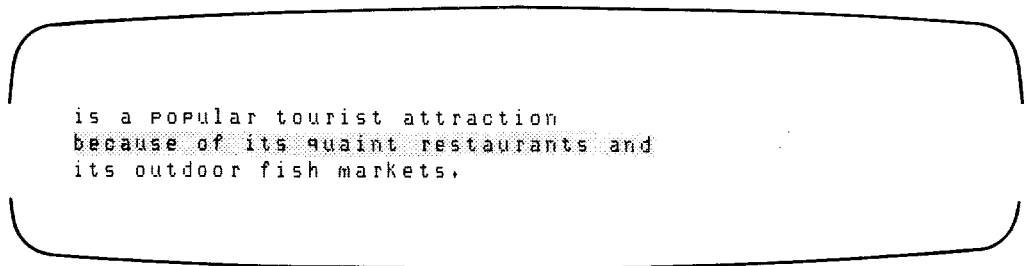
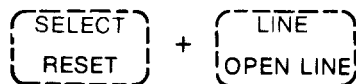
2. Put the contents of the select range into a paste buffer by pressing the CUT function key:



3. Move the cursor to the line that you want to append:



4. Put the line to be appended in a select range:

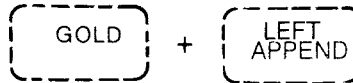


5. Now press the APPEND function key. On VT52s you need to press the GOLD key first, as shown below. The line disappears from the screen and is stored in a buffer:

VT100

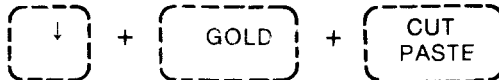


VT52



```
is a popular tourist attraction  
its outdoor fish markets.
```

6. Move the cursor down one line and press the GOLD and PASTE function keys:



```
is a popular tourist attraction  
its outdoor fish markets.  
Castle Island, a small fishing community  
on the coast of Maine,  
because of its quaint restaurants and
```

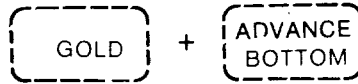
Replacing and Substituting Text

The following two functions, REPLACE and SUBS, are similar. The REPLACE function is necessary for the SUBS operation.

REPLACE

REPLACE deletes the text that you put in a select range and replaces it with the contents of the paste buffer.

Move the cursor to the bottom of the buffer and insert the sentence as shown. The example shows how to replace the word "urban" with the word "industrial":



The large urban city annexes the smaller village. (RET)
[EOB]

Move the cursor to the start of the line:



The large urban city annexes the smaller village.

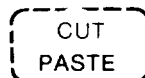
The steps for using REPLACE are as follows:

1. Press the SELECT function key and type "industrial":



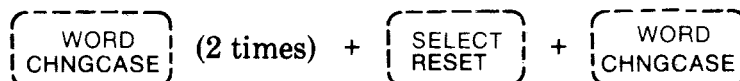
industrial The large urban city annexes the smaller village.

2. Press CUT to store the phrase in the paste buffer:



The large urban city annexes the smaller village.

3. Move the cursor to the word you want to replace and put this word in a select range:



The large urban city annexes the smaller village.

4. Press GOLD and the REPLACE function key. The sentence shows the change you made.

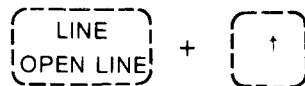
VT100

VT52



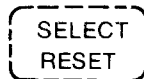
The large industrial city annexes the smaller village.

You can also use the FIND function in REPLACE operations. In the following example, the word "nearby" replaces the strings "large" and "smaller". Move the cursor back to the start of the line for the example:



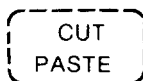
The large industrial city annexes the smaller village.

1. Press the SELECT key and enter some text:



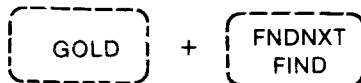
~~nearby~~ The large industrial city annexes the smaller village.

2. Press CUT to place the text in the paste buffer:



The large industrial city annexes the smaller village.

3. Use FIND to locate the text you want to replace:



The FIND function causes a prompt to appear on your screen:

Search for: large ADVANCE
BOTTOM

The cursor moves to the word "large":

The large industrial city annexes the smaller village.

4. Press GOLD and the REPLACE function key to replace the word "large" with the contents of the PASTE buffer:

VT100

VT52

GOLD + APPEND
REPLACE

GOLD + UP
REPLACE

The nearby industrial city annexes the smaller village.

You can repeat the last two steps if you want to replace another word with the contents of the PASTE buffer. Use GOLD and the FIND function key to locate the word "smaller":

GOLD + FNDNXT
FIND → Search for: smaller ADVANCE
BOTTOM

The cursor moves to the word you specified:

The nearby industrial city annexes the smaller village.

Press the GOLD and REPLACE function keys again. The result is as follows:

VT100

VT52

GOLD + APPEND
REPLACE

GOLD + UP
REPLACE

The nearby industrial city annexes the nearby village.

SUBS

SUBS lets you find a string of characters and replace it with the contents of the paste buffer. SUBS works like the REPLACE function, except that the SUBS function automatically includes a "find next" operation.

The next example shows how to change the phrase "woolen mills" to "ship building" in the following frames. Move the cursor to the bottom of the buffer and enter the text:

GOLD + ADVANCE
BOTTOM

Maine, once known for its woolen mills, (RET)
is now a leader in lumber operations. (RET)
Formerly, woolen mills prospered in (RET)
nearly every town on the coast. Today, Bath (RET)
is the primary center for woolen mills. (RET)

To make the change in this example, you place the new phrase in a paste buffer, find the existing phrase, and substitute it throughout the paragraph. Move the cursor to the start of the text for this example:

↑ (5 times)

Maine, once known for its woolen mills,
is now a leader in lumber operations.
Formerly, woolen mills prospered in
nearly every town on the coast. Today, Bath
is the primary center for woolen mills.

1. Press the SELECT function key and enter "ship building" from the keyboard. This puts the words you type in a SELECT range.

SELECT
RESET

~~ship building~~ Maine, once known for its woolen mills,
is now a leader in lumber operations.
Formerly, woolen mills prospered in
nearly every town on the coast. Today, Bath
is the primary center for woolen mills.

2. Press the CUT function key to store "ship building" in a paste buffer. The words disappear from the screen.

CUT
PASTE

Maine, once known for its woolen mills,
is now a leader in lumber operations.
Formerly, woolen mills prospered in
nearly every town on the coast. Today, Bath
is the primary center for woolen mills.

3. When you press GOLD and the FIND function key, you move the cursor to the first occurrence of the phrase "woolen mills". You must use FIND (which places the phrase in a search buffer) in order for subsequent SUBS operations to work.

GOLD + FNDNXT
FIND → Search for: woolen mills ADVANCE
BOTTOM

Maine, once known for its woolen mills,
is now a leader in lumber operations.
Formerly, woolen mills prospered in
nearly every town on the coast. Today, Bath
is the primary center for woolen mills.

When you press the GOLD and SUBS function keys, the change is made and the cursor moves to the next occurrence of the phrase:

GOLD + ENTER
SUBS

Maine, once known for its ship building,
is now a leader in lumber operations.
Formerly, woolen mills prospered in
nearly every town on the coast. Today, Bath
is the primary center for woolen mills.

4. Continue to substitute the phrase throughout the paragraph by pressing GOLD and SUBS. If you want to skip an occurrence of the phrase, press FNDNXT instead of GOLD and SUBS.

Using Line Editing Commands

The two functions in this section, ENTER and COMMAND, allow you to use line editing commands during keypad editing. This capability greatly extends the usefulness of keypad editing. An example of the ENTER and COMMAND functions is shown at the end of this section.

ENTER

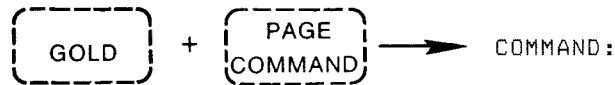
ENTER lets you perform searches and use line editing commands. When you use the FIND function and type a search string, pressing ENTER completes the string. When you press GOLD and the COMMAND function key to type a command, pressing ENTER causes EDT to execute the command.

COMMAND

COMMAND lets you enter EDT command level and line editing commands. Press the COMMAND function key and type in these commands at the keyboard. To execute the commands, press the ENTER function key.

Using COMMAND and ENTER

1. Press the GOLD key and then the COMMAND function key. The COMMAND prompt appears at the bottom of the screen.



2. Type a command level command, which appears after the prompt:

COMMAND: SET NOTRUNCATE

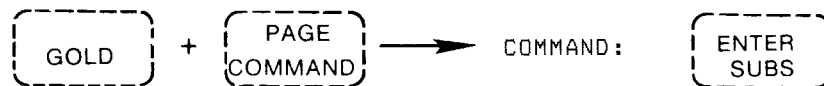
3. Press the ENTER function key to execute the command.

Upon completion of a line editing command, the screen is updated as required and you return to keypad editing. If you enter a command such as EXIT or QUIT and press the ENTER function key, you end the EDT editing session.

Correcting Mistakes When You Enter Commands

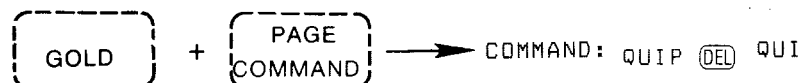
If you make a mistake in the middle of pressing a sequence of function keys, you can correct what you have done with the ENTER function.

For example:

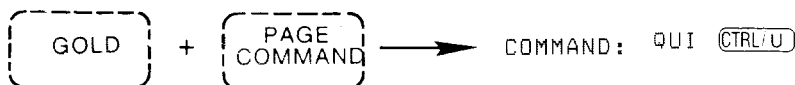


You can delete the text you enter with the COMMAND function two ways. In these examples, assume you mistype the QUIT command:

1. To delete the last character typed, press **DEL**:



2. To delete the entire operation, press **CTRL/U** :



Pressing **CTRL/U** deletes the entire command line and returns the cursor to where it was in the editing session. Note, however, that if you type the "T" in "QUIT" and press the ENTER function key, EDT executes the command.

Neither **DEL** nor **CTRL/U** will undo an operation once it is executed.

Special Characters, Changing Case, and Filling Lines

These functions let you insert special characters, change characters to uppercase or lowercase, and perform a FILL operation on a section of text.

SPECINS

SPECINS lets you insert nonprinting ASCII characters into your text. (See Appendix B for a list of ASCII character code decimal equivalents.) Press the GOLD function key, enter the decimal representation of the ASCII character at the keyboard, and then press the GOLD and SPECINS function keys.

For example, suppose you wanted to insert a line feed character into the file. You would use the following procedure:

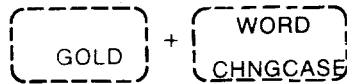
1. Press GOLD and type the ASCII numeric equivalent of the character. Line feed is decimal 10.
2. Press GOLD and then the SPECINS function key.

SPECINS is the only way to insert a carriage return character without having it interpreted as a line terminator.

CHNGCASE

CHNGCASE inverts the case of alphabetic characters; that is, it changes uppercase alphabetic characters to lowercase, and the reverse. Press the GOLD key and the CHNGCASE function key to change the case of the character the cursor is on:

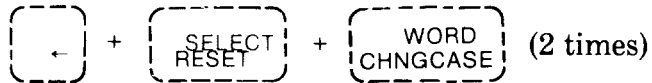
Formerly, woolen mills prospered on



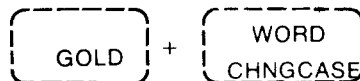
formerly, woolen mills prospered on

Note that the cursor moves one character to the right when you perform a CHNGCASE operation. This happens when the editing status is ADVANCE, which is the default. If EDT is in backup mode, the cursor moves one character to the left.

You can also change the case of text in a select range or a search string. If you press CHNGCASE when the cursor is on a search string or select range, the case of all these characters is changed. Move the cursor to the start of the line for this example:



formerly, woolen mills prospered on

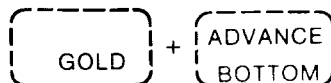


FORMERLY, WOOLEN mills prospered on

FILL


FILL fills a selected range of lines to the limit of the line width. This function exists on the VT100 as a keypad key and as CTRL/F on the VT52.

For example, you may want to limit the width of a section of text to 20 characters. The FILL function allows the maximum number of characters on line without breaking words. Move the cursor to the bottom of the buffer, then enter the text shown to follow the example:




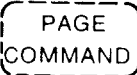
Wolfeboro and Melvin hug the shore of ^(RET)
Lake Winnepesaukee. They are Picturesque ^(RET)
towns with good prospects for light ^(RET)
industry. The main industry ^(RET)
of Wolfeboro and Melvin is tourism. ^(RET)

Move the cursor to the start of the second line for the example:

 (4 times)

Wolfeboro and Melvin hug the shore of
Lake Winnepesaukee. They are Picturesque
towns with good prospects for light
industry. The main industry
of Wolfeboro and Melvin is tourism.

1. Press GOLD and the COMMAND function key:

 +  COMMAND: ?

2. Type "SET WRAP 20" after the COMMAND: prompt to limit the width of the text in the select range. Then press the ENTER function key:

COMMAND: SET WRAP 20



3. Select the range of lines that should not exceed 20 characters in length. Use the SELECT function key and move the cursor to the end of the range of lines:

SELECT
RESET + ↓ (3 times)

```
Wolfeboro and Melvin hug the shore of  
Lake Winnepesaukee. They are picturesque  
towns with good prospects for light  
industry. The main industry  
of Wolfeboro and Melvin is tourism.
```

4. Now use the FILL function:

VT100

VT52

GOLD + SECT
FILL

CTRL F

```
Wolfeboro and Melvin hug the shore of  
Lake Winnepesaukee.  
They are picturesque  
towns with good  
prospects for light  
industry. The main  
industry  
of Wolfeboro and Melvin is tourism.
```

← 20 characters →

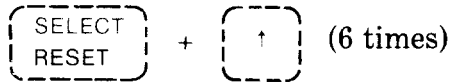
5. To reset the word wrap, press GOLD and the COMMAND function key:



Type SET WRAP 40 after the prompt and press ENTER:



Use a reverse SELECT to restore the lines you previously wrapped to 20:



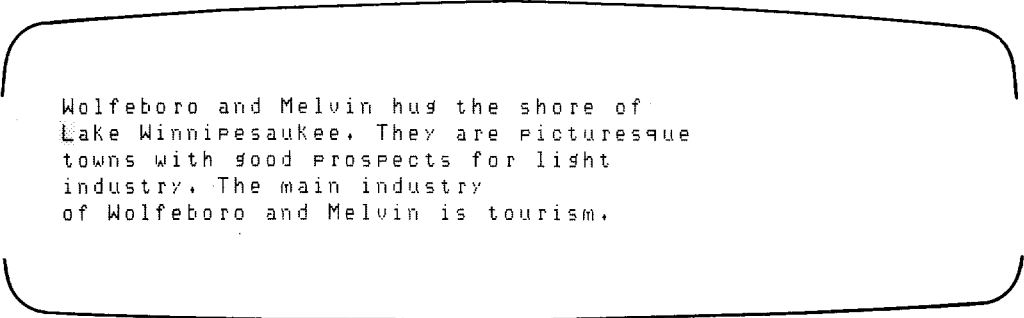
Wolfeboro and Melvin hug the shore of
Lake Winnepesaukee.
They are picturesque
towns with good
prospects for light
industry. The main
industry
of Wolfeboro and Melvin is tourism.

Use the FILL function again to reset the text to the original wrap:

VT100

VT52





Wolfeboro and Melvin hug the shore of
Lake Winnepesaukee. They are picturesque
towns with good prospects for light
industry. The main industry
of Wolfeboro and Melvin is tourism.

Setting Tabs

This section describes how to set tabs in keypad editing. For more information, see the description of the SET command in Chapter 9.

CTRL/A

CTRL/A sets the tab position to the present cursor position. If the present cursor position is not a multiple of the SET TAB number, an error message occurs.

CTRL/E

CTRL/E increases the TAB level one count, where the count is set by the SET TAB command.

CTRL/D

CTRL/D decreases the TAB level one count, where the count is set by the SET TAB command.

CTRL/T

CTRL/T performs a TAB ADJUST operation on a select range. (See Chapter 8 for a description of TADJ.) The characters in the select range move one tab stop to the right, unless you use a minus (–) character or a repeat count. You can use a repeat count for this function by pressing GOLD and a number from the keyboard before pressing the CTRL/T.

NOTE

You can also use the GOLD key on the keypad, instead of the **CTRL** key on the keyboard, for the following functions:

| | |
|--------|--------|
| CTRL/A | CTRL/E |
| CTRL/D | CTRL/U |
| CTRL/Z | CTRL/W |
| CTRL/T | |

For more information on these keys, see the description of the SET TAB command in Chapter 9.

CTRL/K and CTRL/C

CTRL/K

CTRL/K lets you use the DEFINE KEY function. You can assign new functions to any of the keypad keys (except GOLD), and to several CONTROL keys. See Chapter 10 for more information on redefining keys.

CTRL/C

CTRL/C ends the current operation and returns you to the keypad command level. CTRL/C does not affect any of your previous edits. For example, if you press the GOLD key and then decide to press CTRL/C instead of using another function, the operation ceases and the screen remains unchanged. EDT returns to the condition it was in before you pressed the GOLD key, and the message "Aborted by CTRL/C" appears at the bottom left of the screen.

Chapter 6

Line Numbers, Text Buffers, and Ranges

This chapter describes the kinds of line numbers that EDT assigns to the text you edit. Also described are the buffers available when you edit text with EDT and ways to specify ranges of lines within these buffers.

You will need to know the information in this chapter if you want to use EDT for line editing.

Line Numbers

EDT assigns two kinds of line numbers to the text you edit:

1. Those displayed on your terminal. These numbers are useful when you want to specify lines in an editing operation, such as TYPE or DELETE.
2. Those stored internally, which are original line numbers. These numbers help EDT keep track of the text that is originally inserted into the main buffer when you start an editing session. Normally, EDT assigns original line numbers. However, you can assign *fixed line numbers* when you end an editing session. These fixed line numbers become the original line numbers at the next EDT editing session.

Numbers Displayed in Line Editing

EDT assigns line numbers to the lines of all text buffers. These line numbers are displayed in line editing.

When you start EDT and insert text, the text you enter is indented 16 spaces. For example:

DCL/PDS

```
EDIT/EDT LATHROP.DAT(RET)
```

MCR/CCL

```
EDT LATHROP.DAT(RET)
```

```
*INSERT(RET)
```

```
Lathrop, Missouri is one of(RET)  
the true garden spots of the(RET)  
Midwest, with its sprawling(RET)  
cornfields and amber waves(RET)  
of grain.(RET)  
(CTRL Z)
```

```
* (RET)
```

Line numbers are displayed to the left of the text in line editing when you use the TYPE command:

```
*TYPE WHOLE(RET)
```

```
1      Lathrop, Missouri is one of  
2      the true garden spots of the  
3      Midwest, with its sprawling  
4      cornfields and amber waves  
5      of grain.
```

```
[EOB]
```

```
* (RET)
```

The line numbers shown in the preceding example do not become part of the contents of the buffer. Instead, these numbers are displayed so you can use them to specify lines of text.

These line numbers can be anywhere from 0.00001 to 42949.67295. EDT numbers the lines of text in the buffer, starting with 1, in increments of 1.

Resequencing Lines

You can change the sequence of line numbers with the line editing command RESEQUENCE. If you type RESEQUENCE only, EDT renumbers all the lines from the cursor to the end of the buffer. (The lines before the cursor retain their line numbers.) This numbering starts with the current line number and continues in increments of 1. You can also specify a range of lines within the file by entering a range after typing RESEQUENCE.

The SEQUENCE qualifier lets you determine the increments EDT uses to number lines. If you type RESEQUENCE/SEQUENCE: and enter two numbers separated by a colon, EDT numbers the lines in the buffer, starting with n in increments of m.

The following example resequences the contents of the buffer, starting with 10, in increments of 5.

```
* RESEQUENCE/SEQUENCE:10:5(RET)
5 lines resequenced
* Type whole(RET)
    10      Lathrop, Missouri is one of
    15      the true garden spots of the
    20      Midwest, with its sprawling
    25      cornfields and amber waves
    30      of grain.
[EOB]
**
```

Fixed Line Numbers

This section describes an advanced feature which is generally used with compiled programs, and which is only applicable for systems that have RMS. If you do not need this kind of feature, skip this section.

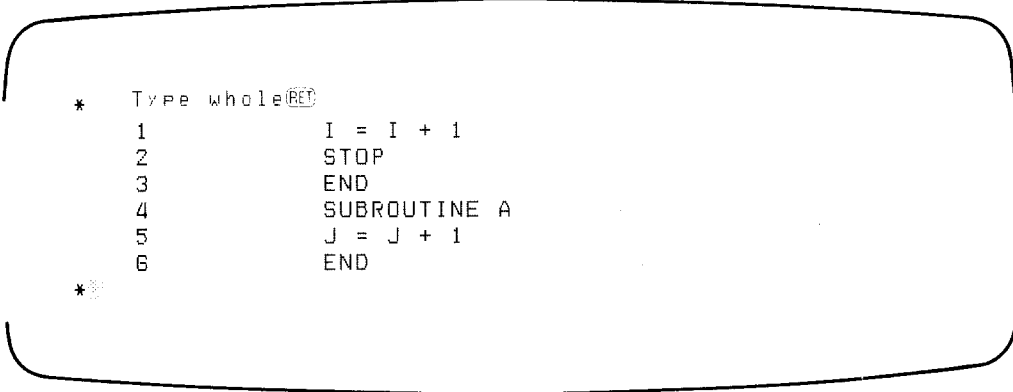
EDT assigns fixed line numbers to keep track of text that is placed in the main buffer when you start an editing session. If the input file consists of VFC (variable with fixed control) records, these fixed line numbers obtained from the VFC fields become the original line numbers. If your system has RMS (Record Management Services), your system documentation contains information about VFC records.

You can associate fixed line numbers with a file when you exit from EDT with the SEQUENCE qualifier. These line numbers are fixed, in that EDT assigns integer line numbers when the file is output.

When you enter EXIT/SEQUENCE:m:n, EDT writes the file in VFC format, placing the fixed line number data in the fixed control portion of the record. (Refer to your system's RMS documentation for more information on VFC files.) When you later start EDT to work on this file, these fixed line numbers become the original line numbers. Additionally, certain language processors (such as FORTRAN-IV-PLUS) display fixed line numbers in the listings of their programs. You can refer to these line numbers to correct diagnostics reported in the compilation.

You do not see fixed line numbers until you request the output of a VFC format file and then compile the program. The fixed line numbers are printed on a listing of the compiled program.

The following example uses a FORTRAN-IV-PLUS program:



```
*      Type whole(RET)
1          I = I + 1
2          STOP
3          END
4          SUBROUTINE A
5          J = J + 1
6          END
**
```

At this point you must exit and use the /SEQUENCE qualifier to create the fixed line numbers:

```
*      EXIT/SEQUENCE:100:10(RET)
```

This command causes EDT to output a VFC format file. The fixed line numbers start with 100 and continue in increments of 10.

When you compile the program, the fixed line numbers are numbered in regular increments. Because the subroutines in some programs duplicate line numbers during compilation, it is useful to have a sequence of fixed line numbers when you get error messages. The following is a listing of the program in VFC format:

```

      .
      .
00100  0001      I = I + 1
00110  0002      STOP
00120  0003      END
      .
      .
00130  0001      SUBROUTINE A
00140  0002      J = J + 1
00150  0003      END
      .
      .

```

Note that the line numbers in the subroutine are the same as the line numbers in the main program. By contrast, the fixed line numbers are distinct and consecutive. You can therefore refer to lines within the program by the fixed line numbers on the listing.

Text Buffers

Two text buffers are created when you start EDT: MAIN and PASTE. EDT creates additional text buffers as you call for them. For example, when you start EDT on a new file, EDT creates the following buffers even though they do not contain text:

```

*   SHOW BUFFER(RET)
=MAIN  0      lines
PASTE  0      lines
*

```

Main Buffer

EDT names the main text buffer MAIN. When you start an editing session, EDT puts the contents of the input file into the main text buffer. EDT transfers the contents of the main text buffer to the output file when you exit the editing session.

Paste Buffer

EDT names the paste text buffer PASTE. The paste buffer is empty at the beginning of each editing session.

The paste buffer stores text from a cut or append operation until the next time you cut or append text. The contents of the paste text buffer are transferred to the current text buffer for paste operations. You can also use PASTE as a text buffer for other editing operations.

Additional Text Buffers

You can use additional text buffers to:

- Retain parts of other buffers
- Include text from other files
- Enter text separately from the main buffer

EDT creates text buffers when you enter a buffer name consisting of 1 to 30 alphanumeric characters. The first character must be alphabetic. You can assign any name to a buffer except MAIN or PASTE.

The examples in this section show the movement of text from existing files into buffers. Unless you use commands to move or delete this text, it remains in the buffers throughout the editing session.

The following example copies lines 10 through 100 of the current text buffer to a text buffer named TEMP. If TEMP does not exist, EDT creates it:

```
* COPY 10 THRU 100 TO =TEMP(RET)
```

The next example copies the contents of NEWFILE.TYP to the paste buffer:

```
* INCLUDE NEWFILE.TYP BUFFER PASTE(RET)
```

In the following example, EDT moves lines 1 through 32 of the text buffer HOLD to the location just ahead of line 88 in the current text buffer.

```
*  MOVE =HOLD 1 THRU 32 TO 88(RET)
```

There is no restriction on the size of any single text buffer except the restriction placed on the set of all text buffers. The maximum size for the set of all text buffers in EDT is approximately 250,000 lines of 80 characters each. The maximum size can be limited by the space restrictions on the system device containing EDT's work space.

The buffer status is displayed through the SHOW BUFFER command (see Chapter 9). The SHOW BUFFER command displays the name of all the text buffers in use during the edit session, the number of lines they contain, and an equal sign (=) preceding the name of the current text buffer.

Range Specifications

Range specifications let you determine the lines on which EDT commands operate. Ranges can specify either single lines or multiple lines. In addition, the multiple line ranges can be contiguous or noncontiguous. (In other words, the numbers in multiple line ranges do not have to be consecutive.) Ranges can be specified for text buffers other than the current text buffer.

To call or to name a text buffer, you enter =name or BUFFER name as a part of the range specification.

Single Line Ranges

A single line range defines the line in a text buffer that you want a command to operate on. Table 6-1 lists the various formats you can use to specify individual lines. Brackets ([]) enclose optional parameters, and the OR symbol (|) separates alternatives.

Table 6-1: Single Line Ranges

| Range | Description |
|-----------------------|--|
| (period) | The line where the cursor is located. |
| number[.decimal] | The line you specify by the number. |
| 'string' "string" | The next line containing the string you specify. If you do not define the search string (that is, you enter a string that does not contain any characters), EDT uses the last search string. |
| -'string' -"string" | The most recent preceding line containing the string you specify. If you do not define the search string (that is, you enter an empty quoted string), EDT uses the last search string. |
| [range] + [number] | The line which is the specified number of lines after the specified range (where range is a single line range and number is an integer). The default range is the current line and the default number is one. |
| [range] - [number] | The line which is the specified number of lines before the specified range (where range is a single line range and number is an integer). The default range is the current line and the default number is one. |
| BEGIN | The first line in the text buffer. |
| END | An empty line following the last line of text in the text buffer. |
| LAST | The last line in the most recent text buffer before the current text buffer. |
| ORIGINAL number | The line numbers assigned to the text in text buffer MAIN from the primary input file. You can locate the text by the original line number even after it has been assigned a new line number. |

You will need to know the information in this chapter if you want to use EDT for line editing.

The following are examples of single line specifications.

```
*  DELETE ,RET
1 line deleted
```

EDT deletes the line where the cursor is located and displays the message "1 line deleted."

```
*  TYPE 12.11RET
    12.11      This is a decimally numbered line.
```

EDT displays line 12.11.

```

*   TYPE - 'Payroll'(RET)
    129           The Payroll records are file 72.a.
*   "(RET)
    247           Payrolls are made up weekly.

```

EDT displays the first line preceding the current line that contains the string "payroll". The next entry is an implied TYPE (empty string) command. EDT searches forward and displays the next occurrence of the string "payroll".

```

*   INSERT .+12(RET)

```

EDT opens the text buffer 12 lines past the cursor location for the insertion of text.

```

*   'ABC' + 3(RET)
    27           This line is three lines later.

```

EDT searches forward for the first occurrence of the string 'ABC', counts forward three lines, and displays the line at that position.

```

*   FIND LAST(RET)

```

EDT moves the cursor to the line last addressed in the text buffer that was used before the current text buffer. The line is not displayed.

```

*   TYPE ORIGINAL 27(RET)
    247           This is the line of interest.

```

EDT displays what it knows as the original line number, which can be the line number it assigned during line editing or a fixed line number. If your file contains fixed line numbers, EDT accepts these as the original line numbers. If not, EDT uses the line numbers that it assigned in the current line editing session.

Suppose you have added text and resequenced the line numbers, and that you now want to change a line of text. You know the original line number but not its present line number. When you enter the TYPE command with the original range specification, EDT displays the line of interest with its current line number.

Contiguous Line Ranges

Contiguous line ranges define sets of consecutive lines within the text buffer. Table 6-2 shows the various formats you can use to display contiguous lines. Brackets ([]) enclose optional parameters, and the OR symbol (|) separates alternatives.

Table 6-2: Contiguous Line Ranges

| Range | Description |
|---|---|
| [range-1]:[range-2] [range-1] THRU [range-2] | The set of lines from range-1 through range-2. The default for either range entry is the current line. Range-1 and range-2 are any single line range specification. |
| [range] # number [range] FOR number | The specified number of lines beginning with the line specified by range (where range is any single line range specification). The default range is the current line. |
| BEFORE | All lines preceding the current line in the current buffer. |
| REST | All lines after and including the current line. |
| WHOLE | The current text buffer. |

The following are examples of contiguous line range specifications.

```
* DELETE 10 THRU 22(RET)
13 lines deleted
```

You could also use a colon instead of THRU:

```
* DELETE 10:22(RET)
13 lines deleted
```

EDT deletes lines 10 through 22 from the current text buffer and displays a message indicating how many lines were deleted. The number of lines from 10 through 22 is determined by the number of lines you have inserted and deleted since the line numbers were assigned.

```
* DELETE BEFORE(RET)
12 lines deleted
```

EDT deletes all of the lines before the cursor position in the current text buffer and displays a message about the number of lines deleted.

Noncontiguous Ranges

Noncontiguous ranges define multiple lines that are not necessarily adjacent to one another. Range specifiers for noncontiguous ranges are:

| | |
|---------------------------|--|
| [range, range, ...] | All lines specified by each range; each range must be a single line. |
| [range AND range AND ...] | |
| [range] All 'string' | All lines in the range containing the specified string. If range is not used, the default is the entire text buffer. |

Some examples are:

```
* MOVE 3,7,10,22,26 TO 60(RET)
*
```

EDT moves the specified lines to the location just before line 60 and renumbers them for that location.

```
* TYPE ALL 'occurrence'(RET)
.
.
.
```

EDT displays all lines in the current text buffer that contain the string "occurrence".

Text Buffer Range Specifications

You can specify ranges in a buffer other than the current text buffer. Type the buffer name before the range specification.

The format for text buffer range specifications is as follows:

| | |
|------------------------|---|
| [= buffer][range] | The default is the current text buffer. |
| [BUFFER buffer][range] | When you use a buffer without a range specification (null range), the default is the entire text buffer, and the cursor is placed at the first line in the text buffer. |

The following are examples of text buffer range specifications:

```
* TYPE =MAIN .(RET)
```

The period in the command is a current line range. EDT enters the text buffer MAIN at the cursor position (current line) when the text buffer was exited and displays the contents of the line at that position.

```
* TYPE BUFFER PASTE ALL /editor/(RET)
```

EDT displays all lines in the text buffer PASTE that contain the string 'editor'.

Chapter 7

Line Editing

This chapter is intended as a reference source of line editing commands.

The line editing prompt (*) is the first character you see when you start EDT. The commands you enter after this prompt affect ranges of lines that you define. (See Chapter 6 for more information on range specifications.)

You can abbreviate most of the commands in this chapter. EDT accepts any input from the minimum abbreviation up to the complete command. The minimum abbreviation of each command is underlined in the command line. For example:

DELETE [range] [/QUERY]

You can type "D" to delete and "Q" to query. EDT also accepts abbreviations that are longer than the minimum, such as "DE" or "DEL" to delete. Abbreviations must be consecutive, however; a command such as "DL" to delete is not valid.

The commands that follow are listed in alphabetical order.

CHANGE

The CHANGE command starts either keypad or nokeypad editing, depending on your terminal type and whether or not you have used the SET KEYPAD or SET NOKEYPAD command. EDT defaults to keypad character editing for VT52 and VT100 terminals and to nokeypad character editing for all other terminals.

The format of the CHANGE command is:

CHANGE [range]

The range lets you specify the line where you want EDT to place the cursor.

To return to line editing, press **CTRL/Z** in keypad editing or use the **EXIT** command in nokeypad editing.

If you type a semicolon after **CHANGE**, the rest of the line is interpreted as nokeypad commands. The following example uses the **TAB** **INSERT** and **EXIT** commands:

```
*  CHANGE;TI EX(RET)
```

Typing **EX** returns you to line editing.

You can use the **SET** command to start nokeypad editing, as follows:

```
*  SET NOKEYPAD(RET)
*  CHANGE(RET)
```

Typing **CHANGE 43** causes EDT to start either keypad or nokeypad editing, and places the cursor on line 43 of the main text buffer:

```
*  CHANGE 43(RET)
```

CLEAR

The **CLEAR** command lets you delete the contents of a text buffer. The format is:

CLEAR buffer

For example, you would clear the contents of the buffer named **EDITOR** as follows:

```
*  CLEAR EDITOR(RET)
```

The **EDITOR** buffer is empty but still exists. Use the **SHOW BUFFER** command if you want to see the results:

```
*  SHOW BUFFER(RET)
EDITOR  0      lines
=MAIN   10     lines
PASTE   2      lines
*
```

You do not need to be editing a buffer to clear its contents. The other buffers are unaffected.

COPY

The COPY command copies text within text buffers. You copy text from one text buffer to another or from one location to another within a text buffer. The text is not deleted from the original location. (To copy text from an external file, use the INCLUDE command.)

The COPY command has the form:

COPY [range-1] TO [range-2] [/QUERY] [/DUPLICATE:n]

EDT copies the set of lines you specify by range-1 to the location in front of the line you specify with range-2. Range-2 is a single line range; the default for range-2 is the current line. Range-2 can be contained in range-1. The QUERY qualifier lets you verify each line to be inserted in the range. DUPLICATE lets you insert the range of text more than once.

If the destination (range-2) is not in the current text buffer, you specify the name of the receiving text buffer (=buffer) immediately after TO. You must give the full name of the text buffer; do not abbreviate. When your source (range-1) is not the current text buffer, you name the source text buffer (=buffer) immediately before range-1.

When you use the QUERY qualifier, EDT prompts you for verification of each line of the range in the COPY command. The prompt is a question mark (?). You have four valid responses to the QUERY prompt:

Y (Yes) Copy this line to range-2.

N (No) Do not copy this line to range-2.

A (All) Copy all remaining lines in range-1 to range-2.

Q (Quit) Quit the copy operation.

The first line of range-2 becomes the current line after completion of the COPY command.

Examples

```
* COPY 15 THRU 60 TO 95(RET)
```

EDT copies lines 15 through 60 of the current text buffer to the position just before line 95 of the current text buffer.

```
* COPY =MAIN TO =TEXT /QUERY(RET)
1          This is the first line to be copied.
?
```

EDT copies the contents of the text buffer MAIN to the buffer TEXT one line at a time. EDT shows each line to be copied followed by the ? prompt. You must respond with one of the prompt answers (Y, N, A, or Q) before EDT continues.

```
* COPY BEGIN TO 65 /DUPLICATE:9(RET)
```

EDT copies the first line of the current text buffer and places it just before line 65 of the current text buffer. This operation is repeated 9 times.

DEFINE KEY

The DEFINE KEY command lets you assign functions to keys that you will use in keypad editing. For more information on redefining keys, see Chapter 10.

DEFINE MACRO

The DEFINE MACRO command assigns a name to a sequence of editor commands.

The DEFINE MACRO command has the form:

DEFINE MACRO macro-name

The DEFINE MACRO command assigns a name to a text buffer. You can enter the text buffer and insert any number of EDT commands. End your entries with a CTRL/Z. The macro-name is a line editing command and becomes a part of the EDT command list for the duration of this editing session. You can define the macro-name to be the same as an existing line editing command, but then that command is no longer available. For example, assume that you want to insert a block of text in a number of places in the text buffer. You create the macro containing the text and name the macro INSERT. Now when you enter INSERT as a command, EDT will execute the contents of the macro INSERT.

You enter the macro-name in response to the line editing prompt (*). EDT makes a search of the EDT command list for every command you enter before it processes the command. When a macro and a command have the same name, EDT redefines the command as the macro.

You can have macros address other macros to a depth limited only by the available memory.

Examples

```
10          This is a line in the text buffer.
* DEFINE MACRO FORM(RET)
* FIND=FORM(RET)
* INSERT(RET)
          SET TAB 4(RET)
          SET ENTITY SENTENCE ';' (RET)
          SET NOWRAP(RET)
          SET NOTRUNCATE(RET)
          FIND LAST+1(RET)
          INSERT; THIS IS A SAMPLE OF THE NEW PROGRAM(RET)
          (CTRL Z)
* FIND=MAIN.(RET)
* (RET)
10          This is a line in the text buffer.
*
```

At line 10 of the current text buffer you determine a need for a macro. The DEFINE MACRO command defines FORM to be a valid EDT command for the duration of this edit session. A text buffer is assigned the name FORM. You transfer EDT to the text buffer FORM with the FIND command and you enter the command string into the text buffer FORM. At the completion of the command entries, you exit the insert mode with a (CTRL/Z). The FIND=MAIN. command returns you to your position in the text buffer prior to the DEFINE MACRO command. Enter a (RET) and EDT displays the current line, line 10.

When you enter FORM in response to the command level prompt, the commands in the text buffer FORM are executed:

The first tab indent is set to 4.

The delimiter for the sentence entity is set to ;.

The NOWRAP parameter is set.

The NOTRUNCATE parameter is set.

EDT locates the last line previous to entering the FORM command and enters this message on the next line:

```
'THIS IS A SAMPLE OF THE NEW PROGRAM'
```

DELETE

The DELETE command deletes lines of text specified by the range.

The DELETE command has the form:

DELETE [range] [/QUERY]

When you do not specify the range, EDT defaults to the current line. When you delete a range of lines, EDT deletes the lines and displays a message stating the number of lines deleted. The message is followed by a display of the next line in the text buffer.

When you use the QUERY qualifier, EDT prompts you for verification of each line of the range in the DELETE command. The prompt is a question mark (?). You have four valid responses to the QUERY prompt:

- Y (Yes) Delete this line.
- N (No) Do not delete this line.
- A (All) Delete all remaining lines in the specified range.
- Q (Quit) Quit the delete operation.

Examples

```
* DELETE(RET)
1 line deleted
   21      The line preceding this line was deleted.
*
```

EDT deletes the line at the cursor position and displays the contents of the next line, line 21.

```

* DELETE 25(RET)
1 line deleted
  26      There are no lines between line 25 and 26.
*

```

EDT deletes line 25 of the current buffer and displays the contents of the next line, line 26.

```

* DELETE BEFORE(RET)
10 lines deleted
  11      There are no decimal numbers in the range.
*

```

EDT deletes all lines preceding the current line and displays the contents of the current line, line 11.

```

* DELETE 12 THRU 42/QUERY(RET)
  12      This is the first line of the range.
?  N(RET)
  13      This is the second line of the range.
?  Y(RET)
1 line deleted
  14      This is the third line of the range.
?  A(RET)
29 lines deleted
  43      This line is not in the range.
*

```

EDT displays each line of the range and the ? prompt.

You must enter one of the prompt QUERY responses before EDT continues to the next line or exits the delete operation.

EXIT

The EXIT command ends an editing session. EDT transfers the contents of the main text buffer to the file you specify.

The EXIT command has the form:

EXIT [file name] [/SEQUENCE[:initial[:increment]]] [/SAVE]

You define the file name for the contents of the main buffer in either the command line or the EXIT command. When you define the file name in the EXIT command, that name takes precedence.

When you use the SEQUENCE qualifier, EDT assigns integer line numbers before the text transfer and places them in a fixed field in the file. These are fixed line numbers and are a part of the file. You define the starting line number through the initial qualifier and the increment between line numbers through the increment qualifier.

The SAVE qualifiers saves the journal file. The journal file is file name.JOU, where the file name is specified in the command line on the output file name. For information on the journal file, see Chapter 3.

Examples

```
* EXIT(RET)
```

EDT transfers the contents of the main text buffer to the file you defined in your command line. EDT returns you to system command level.

```
* EXIT NAME.NEW/SEQUENCE:5:5/SAVE(RET)
```

EDT transfers the contents of the main text buffer, with line numbers, to your file NAME.NEW. The number of the first line in the file is 5 and each of the following lines is incremented by 5. The editor saves the journal file.

FIND

The FIND command locates the line specified by range. EDT does not display the line located.

The FIND command has the form:

FIND range

Range is a single line range. Use the FIND command to move between text buffers. EDT locates the selected range, positions the cursor at the start of that line, and then displays the command prompt.

Examples

```
* FIND 45(RET)
*
```

EDT finds line 45 of the current text buffer. EDT displays the * prompt when the line is found.

```
* FIND =STORE 45(RET)
*
```

EDT finds line 45 of the text buffer STORE. EDT displays the * prompt when the line is found.

HELP

The HELP command displays information on requested topics.

The HELP command has the form:

HELP [topic [subtopic]]

A topic or subtopic can have one of the following formats:

- An alphanumeric string (for example, a command line or a qualifier)
- The wild card or match-all symbol (*)

To obtain a list of the valid topics, enter **HELP (RET)**.

The subtopics available on a topic are listed at the end of each topic.

Examples

```
* HELP SUBSTITUTE NEXT(RET)
```

EDT displays the HELP text for the SUBSTITUTE NEXT command.

```
* HELP CHANGE SUBCOMMAND(RET)
```

EDT displays the HELP text for CHANGE subcommands.

* HELP^(RET)

EDT slowly displays, or scrolls, all help text alphabetically. If you enter ^(CTRL/S), EDT stops scrolling. Enter ^(CTRL/Q) to restart scrolling.

INCLUDE

The INCLUDE command copies files into text buffers. Lines added through the INCLUDE command are numbered by EDT to ensure that their position immediately precedes the specified range.

The INCLUDE command has the form:

INCLUDE file name [range]

The file name is the name of the file you want to copy.

EDT copies the specified file to the current text buffer immediately in front of the first line of [range]. The transfer does not modify the source file. When the INCLUDE operation is complete, EDT displays the command prompt.

Examples

* INCLUDE TEXT.DAT^(RET)

EDT copies the contents of the file named TEXT.DAT into the current text buffer just in front of the present line.

* INCLUDE TEXT.FIL =TEXT^(RET)

EDT copies the contents of the file name TEXT.FIL into the buffer you have named TEXT. EDT copies the text in front of any text in the text buffer.

* INCLUDE FILE.TEN =ELEVEN 60^(RET)

EDT copies the contents of the file named FILE.TEN into the text buffer you have named ELEVEN. EDT loads the text into the text buffer ELEVEN just in front of line 60.

INSERT

The INSERT command lets you insert text into a buffer. EDT numbers the lines of text that you insert to ensure that their position immediately precedes the specified range.

The INSERT command has the form:

`INSERT [range] [;line to be inserted]`

To exit the INSERT command, enter `CTRL/Z`.

When you use the range qualifier, EDT inserts the text you enter before the first line of the specified range. Range is a single line range. When you do not use the range qualifier, EDT inserts the text you enter just before the current line.

When you end the INSERT command with a semicolon, you can use the rest of the command line (until you press `RET`) to insert text. EDT inserts the text at the specified range and exits the insert operation when you press `RET`. When you do not use the semicolon, the insert operation continues until you enter a `CTRL/Z`.

Examples

```
*   INSERT 12RET
```

EDT allows you to insert text just before line 12. Enter as many lines of text as you wish. EDT assigns decimal line numbers to all lines entered. EDT remains in the insert mode until you enter a `CTRL/Z`.

```
*   INSERT =TEXT 30;Enter new employee numbersRET
*
```

EDT inserts "Enter new employee numbers" just before line 30 in the buffer named TEXT. When EDT completes your insert, the line editing prompt is displayed.

MOVE

The MOVE command moves the lines defined by one range to the location preceding the line specified by another range. The copied lines are numbered by EDT to ensure that their position immediately precedes the specified range.

When you use the MOVE command, EDT deletes the original copy of the text. Use the COPY command to copy without deleting the original text.

The MOVE command has the form:

MOVE [range-1] TO [range-2] [/QUERY]

Range-1 is the text you wish to move. Range-2 is the destination for the text of range-1 and becomes the current line on completion of the move. If you omit either range-1 or range-2, the default for the omitted range is the current line. When the move is complete, EDT displays a command line prompt.

When you use the QUERY qualifier, EDT prompts you for verification of each line of range-1 in the MOVE command. The prompt is a question mark (?). You have four valid responses to the QUERY prompt:

- Y (Yes) Move this line to range-2.
- N (No) Do not move this line to range-2.
- A (All) Move all of the remaining lines in range-1 to range-2.
- Q (Quit) Quit the operation.

Examples

```
* MOVE TO 65(RET)
```

EDT moves the current line to line 65.

```
* MOVE =TEXT WHOLE TO 47(RET)
```

EDT copies the contents of your text buffer TEXT to the position just in front of line 47 of the current text buffer and then deletes the contents of text buffer TEXT. EDT assigns new line numbers to the moved lines.

```
* MOVE 24 THRU 88 TO =MAIN 60(RET)
```

EDT copies lines 24 through 88 of the current text buffer to the point just in front of line 60 in the text buffer MAIN and then deletes those lines in the current buffer. EDT assigns new line numbers to the moved lines.

Null (Implied TYPE)

The null command performs an implied TYPE. In other words, when you press (RET) immediately after the asterisk prompt, EDT displays the next line of text.

You can insert a range specification before pressing **(RET)**. However, the ranges REST, WHOLE, BEGIN, END, LAST, and ALL must be preceded by a percent sign (%) when used with the null command.

The null command has the form:

[range] **(RET)**

Examples

* **- (RET)**

The minus sign (—) reverses the direction so that EDT displays the line preceding the current line.

* **15 THRU 60 (RET)**

EDT displays each line in the range in sequence.

* **%REST (RET)**

EDT displays the rest of the text in the current text buffer.

PRINT

Use the PRINT command to copy text from a text buffer into a file. EDT puts the transferred text into a file that can be printed onto a listing.

The PRINT command has the form:

PPRINT file name [range]

Use the range to select a portion of the text buffer to be copied to the file in page printable format. When you do not make a range entry, the default is the current text buffer.

When the transfer is complete, EDT returns you to your last position in the current text buffer and displays the command level prompt.

Examples

* **PRINT YOUR.FIL (RET)**

EDT copies the contents of the current text buffer to a file named YOUR.FIL, which can be printed.

```
* PRINT THIS.FIL=TEXT 1 THRU 86(RET)
```

EDT copies lines 1 through 86 of your text buffer named TEXT to a file named THIS.FIL, which can be printed.

QUIT

The QUIT command ends the current editing session without saving the contents of the main text buffer.

The QUIT command has the following form:

```
QUIT [/SAVE]
```

You use the SAVE qualifier to save the journal file. For more information about the journal file, see Chapter 3.

Examples

```
* QUIT(RET)
```

EDT returns you to system command level and saves nothing from the current session.

```
* QUIT/SAVE(RET)
```

EDT saves the contents of the journal file under the name you defined in the command line. EDT returns you to system command level.

REPLACE

Use the REPLACE command to delete the lines specified by the range and to add new text. When the lines are deleted, EDT displays a message stating the number of lines deleted and then enters insert mode. The text buffer allows the insertion of new text at the first line in the range specification.

The REPLACE command has the form:

```
REPLACE [range] [;line to be inserted]
```

To exit the REPLACE command, enter ^(CTRL/Z).

When you do not specify a range, EDT deletes the current line and lets you replace it with one or more lines of new text. EDT renumbers the inserted text, using the numbers of the deleted lines. When the text you insert exceeds the line count of the deleted text, EDT assigns decimal line numbers to ensure that the new line numbers do not conflict with line numbers outside the specified range. When there are more deleted lines than inserted lines, the excess line numbers are not used.

To replace the specified range with a single line of text, you end the REPLACE command with a semicolon and enter the new text on the same line. When you press **RET** to end the line, EDT displays the command line prompt.

To return to the asterisk prompt when you do not use the semicolon, enter a **CTRL/Z**. EDT moves the cursor to the line following the last line deleted.

Examples

```
* REPLACE 3 THRU 12RET
10 lines deleted
```

EDT deletes lines 3 through 12 of the current text buffer and allows you to insert new text at line 3. When you complete your insertion, press **CTRL/Z**.

```
* REPLACE 9; Delete and replace this line.RET
```

EDT deletes line 9 and inserts the text between the semicolon and the **RET** as the new line 9.

RESEQUENCE

The RESEQUENCE command assigns new line numbers to the contents of a buffer.

The RESEQUENCE command has the form:

RESEQUENCE [range] [**/SEQUENCE** [:initial[:increment]]]

The lines you specify for resequencing by the range specification must be contiguous. When you do not specify a range, EDT resequences all lines in the current text buffer. When you use the sequence qualifier, EDT sets the first line to the initial value and increments the succeeding line numbers with the value of increment. When the lines are resequenced, EDT displays the line editing prompt and a message telling how many lines are resequenced.

If you do not use the SEQUENCE qualifier, EDT sets the increment to 1, and the initial number is the first line of the specified range.

EDT prevents the assignment of nonsequential or duplicate line numbers.

Examples

```
* RESEQUENCE 22 THRU END(RET)
```

EDT resequences the line number of the current buffer starting with line 22. EDT increments the line numbers by a count of 1.

```
* RESEQUENCE =TEXT /SEQUENCE:10:10(RET)
```

EDT resequences the line numbers of the contents of your buffer named TEXT. The first line number is 10, and EDT increments the line numbers by 10.

SUBSTITUTE

The SUBSTITUTE command replaces occurrences of one specified string with another string. A string can be from 0 to 64 characters.

The SUBSTITUTE command has the form:

```
SUBSTITUTE /string-1/string-2/ [range] [/BRIEF[:n]] [/QUERY] [/NOTYPE]
```

Any nonalphanumeric character, such as slash (/), can be used as a string delimiter, provided you use the same delimiter in all places. For example, EDT will accept either of the following:

```
* SUBSTITUTE/Lathrop/Littleton/RET
```

```
* SUBSTITUTE$Lathrop$Littleton$RET
```

EDT will not accept mixed delimiters, such as:

```
*SUBSTITUTE$Lathrop/Littleton%
```


You cannot use a character as a delimiter if the character appears in string-1 or string-2. For example, you will not get the correct results if you type the following, because a dollar sign appears in one of the strings (\$5.00):

```
*SUBSTITUTE$Send $5.00$Mail me cash$
```

If you do not specify a range, EDT replaces the first occurrence of the string with the substitute string. When you specify a range, EDT replaces all the occurrences of string-1 within the range with string-2. EDT displays the line after the substitution is made. EDT returns you to the first line in the specified range at the end of the substitution.

When you use the BRIEF:n qualifier, EDT displays the first n characters of the lines containing string-1. The default for n is 10.

When you use the QUERY qualifier, EDT precedes each substitution with a question (?) prompt. The responses you can give are:

Y (Yes) Substitute this string.

N (No) Do not substitute this string.

A (All) Make all of the remaining substitutions without further queries.

Q (Quit) Quit the substitution operation.

When you use the NOTYPE qualifier, the lines where EDT makes the substitutions are not displayed.

Examples

```
* SUBSTITUTE /used/Pregowned/WHOLE/QUERY(RET)
```

EDT searches the current buffer for the word "used". For each occurrence, EDT prints the line containing "used", followed by a question mark (?). You read the line to determine the validity of making the substitution. The string delimiter used in this example is the same as the qualifier delimiter.

```

      22          This is a command used for substitutions.
? N(RET)
      46          It can be used for a range of lines.
? N(RET)
No substitutions
*
```

After you respond to the ? prompt with N, EDT continues searching the text buffer for occurrences.

When you respond Y, EDT replaces “used” with “preowned”. If there is more than one occurrence of “used” in the line, the process is repeated. On each occurrence of “used” in the line, EDT displays the corrected line and, after the last substitution, the number of substitutions made.

```

      20          The word list appears once in this sentence.
* SUBSTITUTE #list#table#/BRIEF:20(RET)
      20          The word table appea
1 substitution made
*
```

EDT searches the current line for the word “list”, replaces it with “table” and displays the first 20 characters of the corrected line and the number of substitutions made. The string delimiter used differs from the qualifier delimiter.

```
* SUBSTITUTE @section@chapter@WHOLE/QUERY(RET)
```

EDT searches the current buffer for the word “section” and then displays in sequence each line containing the word “section”, followed by a question mark (?). When you respond with one of the four QUERY qualifiers, EDT continues. When your response is a Y, the editor replaces “section” with “chapter” and searches for the next occurrence of “section”. The string delimiter used differs from the qualifier delimiter.

SUBSTITUTE NEXT

The **SUBSTITUTE NEXT** command replaces the next occurrence of a specified string with another string. The cursor remains at the line where the substitution is made.

The **SUBSTITUTE NEXT** command has the form:

[SUBSTITUTE] NEXT [/string-1/string-2/]

EDT searches for the next occurrence of string-1 from the current location forward. When EDT makes a substitution, the line in which the substitution is made becomes the current line.

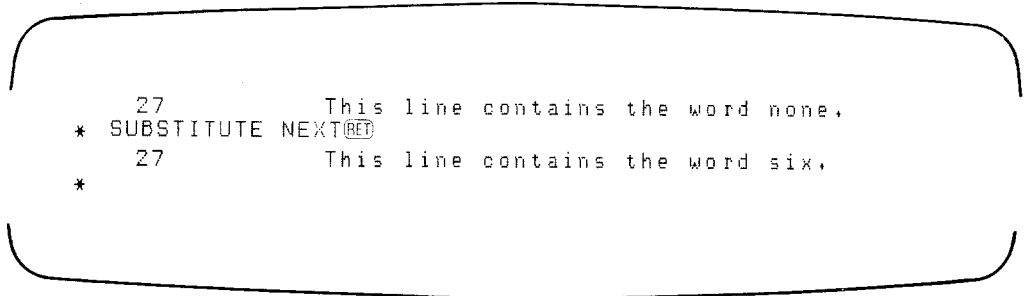
If you specify neither string-1 nor string-2, EDT uses the strings specified by your last **SUBSTITUTE** command.

Examples

Assume that your last substitute command was:

```
* SUBSTITUTE/none/six/(RET)
```

You wish to find the next occurrence of the word "none" to replace it with "six".



```
27 This line contains the word none.  
* SUBSTITUTE NEXT  
27 This line contains the word six.  
*
```

EDT searches for the next occurrence of your last **SUBSTITUTE** command string-1 and replaces it with string-2. EDT replaces the word "none" with "six" and displays the corrected line.

```

* NEXT/reason/EMOTION/RET
  43          This line now contains the word EMOTION.
*

```

EDT searches for the next occurrence of “reason” and replaces it with “EMOTION”. EDT displays the corrected line. Uppercase and lowercase characters are equivalent when matching, but substitution uses the same case as the command.

TYPE

The TYPE command displays a specified range of lines.

The TYPE command has the form:

```

TYPE [range] [/BRIEF[:n]] [/STAY]

```

The range you select can be a single line or multiple lines. The multiple lines do not have to be contiguous. The first line you specified in the range becomes the current line.

When you select the BRIEF qualifier, EDT displays the first n characters of the selected lines. The default for n is 10 characters.

When you select the STAY qualifier, EDT does not change the cursor position.

Examples

```

  30          There are two lines left in the current buffer.
* TYPE RESTRET
  30          There are two lines left in the current buffer.
  31          This is the last line.
[EOB]
*

```

EDT displays the remainder of the current text buffer from the current line. The first line displayed is the current line.

```
* TYPE 27(RET)  
  27          Line 27 is for display.  
*
```

EDT displays line 27, the current line.

```
* TYPE 3,5,6,9,24(RET)  
  3          This is line 3.  
  5          This is line 5.  
  6          This is line 6.  
  9          This is line 9.  
 24          This is line 24.  
*
```

EDT displays lines 3, 5, 6, 9, and 24. Line 3 is the current line.

```
* TYPE 26#3(RET)  
 26          This is line 26.  
 27          This is line 27.  
 28          This is line 28.  
*
```

EDT displays three lines, beginning with line 26.

```
* TYPE 40:END ALL 'news'(RET)  
 41          This is the first line after 40 containing news.  
 44          This is the last line after 40 containing news.  
*
```

EDT displays all lines, beginning with line 40, that contain the string 'news'.

```

* TYPE =TEXT 'Journal'(RET)
  33          Line 33 contains the word Journal.
*

```

EDT searches your buffer TEXT for the word “journal” and then displays the first line containing “journal”.

```

      32          This is the current line.
* TYPE 1 THRU 15 /STAY(RET)
  1          This is line one.
  .
  .
  .
  15          This is line fifteen.
*

```

EDT displays lines 1 through 15 of the current text buffer. The cursor remains at line 32.

WRITE

The WRITE command copies the defined range of text from a text buffer to the specified file. The WRITE command does not change the contents of the text buffer.

The WRITE command has the form:

WRITE file name [range] [/SEQUENCE[:initial[:increment]]]

When you specify a range, EDT copies that text to the file. When you do not specify a range, EDT copies the contents of the current text buffer to the file.

When you use the SEQUENCE qualifier with WRITE, EDT writes the line numbers as a fixed field in the file. In other words, these fixed line numbers become part of the file.

Examples

```
* WRITE PART.FIL 10 THRU 90(RET)
```

In the preceding example, EDT transfers a copy of the contents of lines 10 through 90 to the file PART.FIL. When the transfer is complete, EDT displays the complete name of the new file, the number of lines written to the file, and the command prompt.

```
* WRITE NEW.DAT =TEXT1/SEQUENCE:10:10(RET)
```

EDT transfers a copy of the text buffer TEXT1 to your file, NEW.DAT. EDT renumbers the lines (beginning with line 10 and incremented by 10) and places them in a fixed field in the file. When the transfer is complete, EDT displays the complete name of the new file, the number of lines written to the file, and the command prompt.

Chapter 8

Nokeypad Editing

This chapter is intended as a reference source of nokeypad commands.

Nokeypad editing is useful if you have a terminal with a video screen but no keypad. With nokeypad editing, you enter all text and commands from the keyboard and display sections of text on the screen. Therefore, you can see how the edits you make affect the contents of a buffer.

Nokeypad editing has essentially the same capabilities as keypad editing, although they are two distinct kinds of editing. Whereas you press keys in keypad editing to use most functions, you must type nokeypad commands.

To start nokeypad editing with a VT52 or VT100 terminal, enter the SET NOKEYPAD and CHANGE commands in line editing. If you do not have a VT52 or VT100, typing SET NOKEYPAD is not necessary:

DCL/PDS

```
␣ EDIT/EDT REPORT.DAT␣
```

MCR/CCL

```
␣ EDT REPORT.DAT␣
```

```
* SET NOKEYPAD␣  
* CHANGE␣
```

The screen goes blank and text from the file you specify (REPORT.DAT) is displayed. The screen is blank if you are creating a file. The cursor is at the upper left of the screen.

When you begin to enter nokeypad commands, the cursor jumps to the bottom of the screen. When you press **(RET)** to complete the command, the cursor returns to its place in the text buffer.

Entities of Text

An entity is a quantity of text that EDT recognizes as a unit. You can use entities in many of the nokeypad commands. For example, the following command tells EDT to delete the entity PAR, or paragraph:

CUTPAR**(RET)**

Table 8-1 describes the entities you can use in nokeypad editing.

Table 8-1: Entities of Text

| Entity | Description |
|------------------------|--|
| C (Character) | C is a single character. The characters include all of the alphabetic, numeric, and special characters on the keyboard. |
| W (Word) | W is a string of characters delimited by one of a set of delimiter characters, including spaces. The default word terminators are spaces, (RET) , tabs, and (FF) (form feed). You can define the word delimiters with the SET ENTITY command. Each delimiter other than space is a word. |
| BW (Beginning Of Word) | BW is the string of characters from the cursor position to the beginning of the word. When the cursor is at the beginning of a word, BW is the previous word. |
| EW (End Of Word) | EW is the string of characters from the cursor position to the end of the word. This includes the character at the cursor position. |
| L (Line) | L is a single line of text, where line is a string of characters between line terminators. |
| BL (Beginning Of Line) | BL is the string of characters from the cursor position to the beginning of the line. When the cursor is at the beginning of a line, BL is the previous line. |
| EL (End Of Line) | EL is the string of characters from the cursor position to the end of the line. This includes the character at the cursor position. |

(Continued on next page)

Table 8-1: Entities of Text (Cont.)

| Entity | Description |
|-------------------------------|---|
| NL (Next Line) | NL is the string of characters from the cursor position to the beginning of the next line. |
| PAR (Paragraph) | PAR is a user-defined entity. EDT defaults to two successive carriage returns for a paragraph delimiter. You define the paragraph delimiter with the SET ENTITY command. |
| BPAR (Beginning Of Paragraph) | BPAR is the string of characters from the cursor position to the beginning of the paragraph. If you are already at the beginning of a paragraph, BPAR selects the entire previous paragraph. |
| EPAR (End Of Paragraph) | EPAR is the string of characters from the cursor position to the end of the paragraph (not including the paragraph delimiter). This includes the character at the cursor. |
| SEN (Sentence) | SEN is a user-defined entity. EDT defaults to period (.), a question mark (?), or an exclamation mark (!) for sentence delimiters. You define the sentence delimiters through the SET ENTITY command. |
| BSEN (Beginning Of Sentence) | BSEN is the string of characters from the cursor position to the beginning of the sentence. If you are at the beginning of a sentence, BSEN selects the entire previous sentence. |
| ESEN (End Of Sentence) | ESEN is the string of characters from the cursor position to the end of the sentence (not including the sentence delimiter). |
| PAGE | PAGE is a user-defined entity. The default for a page is the text between two form feed characters (including the second form feed). You define the page delimiters with the SET ENTITY command. |
| BPAGE (Beginning Of Page) | BPAGE is the text from the cursor position to the beginning of the page. If you are already at the beginning of a page, BPAGE selects the entire previous page. |
| EPAGE (End Of Page) | EPAGE is the text from the cursor position to the end of the page (not including the page delimiters). |
| BR (Beginning Of Range) | BR is the string of characters to the left of the cursor in the current text buffer. This is equal to a BL entity plus all preceding lines. |
| ER (End Of Range) | ER is the string of characters to the right of the cursor in the current text buffer. This is equal to an EL entity plus all remaining lines. |
| V (Vertical) | V is equivalent to L except that the cursor stays in the same column. |

(Continued on next page)

Table 8-1: Entities of Text (Cont.)

| Entity | Description |
|-------------------|--|
| 'string' | A string is a group of characters within quotation marks. A string is used in search operations. When you specify a string with the D (delete) or CUT command, EDT locates the string in the buffer and deletes it. |
| SR (Select Range) | SR is the text between the last SEL (select) command and the cursor position. When select range is not active (no select command entered) and the cursor is at the present search string, SR selects the search string. When neither of the above conditions exists, SR applies to the CHGC (change case) command. For the CHGC command, SR selects a single character in the current direction. |

No keypad Command Structure

The no keypad commands have one and only one form (no abbreviations are allowed). You can put no keypad character editing commands together in a series; no delimiter is required between the commands but one or more spaces are allowable.

The no keypad character editing commands consist of one or more of the following elements in a specified sequence:

- Command** The command specifies EDT's action.
- Count** The count specifies the number of entities the command acts upon. The count must be from 1 to 32767 decimal. If you do not specify a count, EDT defaults to 1.
- Direction** The plus sign (+) indicates the action performed by the command is forward (to the right or down from the cursor). The minus sign (-) indicates that the action performed by the command is backward (to the left or up from the cursor).

EDT defaults to the current direction. When you start an editing session, EDT commands work forward in the buffer until you change the direction.
- Entity** The entity defines the basic element of text (for example, character, word, or paragraph).
- Buffer** The buffer defines a location to store text and is used in the CUT, PASTE, and APPEND operations. When you start an editing session, EDT places text into the main buffer.

You can repeat any string of nokeypad commands by enclosing the string with parentheses and preceding the parentheses with a count. When a command in the string fails, EDT exits the repeated string.

Nokeypad Commands

The commands and command formats in this section are listed in alphabetical order. Brackets ([]) show optional parameters that you can specify. The OR symbol (|) signifies a choice of options. For example, [+|-] means that EDT can perform an operation forwards (+) or backwards (-) through the buffer.

ADV (Advance)

The ADV command sets all commands forward (the defined operation is to the right and down from the cursor position) unless you override ADV with a minus sign (-). The format is:

ADV

APPEND

The APPEND command moves the specified entities to another text buffer. The format is:

[+|-][repeat-count]APPEND[+|-][entity-count][+|-]entity[=buffer]

EDT deletes the moved text from the current text buffer. Use the =buffer range specification to name the receiving text buffer; EDT defaults to the paste text buffer. EDT appends the specified entities to the end of the contents of the receiving text buffer. See the example in the CUT command description.

ASC (ASCII)

The ASC command causes EDT to display an ASCII character when you specify the character's decimal number representation. The format is:

[count]ASCII

EDT ignores a minus sign (-) preceding the count. See Appendix B for the ASCII code decimal equivalents.

Example

```
This is line one.  
This is line two.  
.  
.  
.
```

10ASC(RET)

```
This is line one.  
This is line two.  
<LF>.  
.  
.
```

The cursor location is just after line two. When you enter 10ASC and (RET), EDT enters <LF> at the cursor position.

BACK (Backup)

The BACK command sets all commands backward (to the left or up from the cursor position). You can override BACK with a plus sign (+) that precedes another command, as in +APPEND entity. The format is:

BACK

CHGC (Change Case)

The CHGC command lets you change the case of the characters within an entity. The format is:

CHGC[entity]

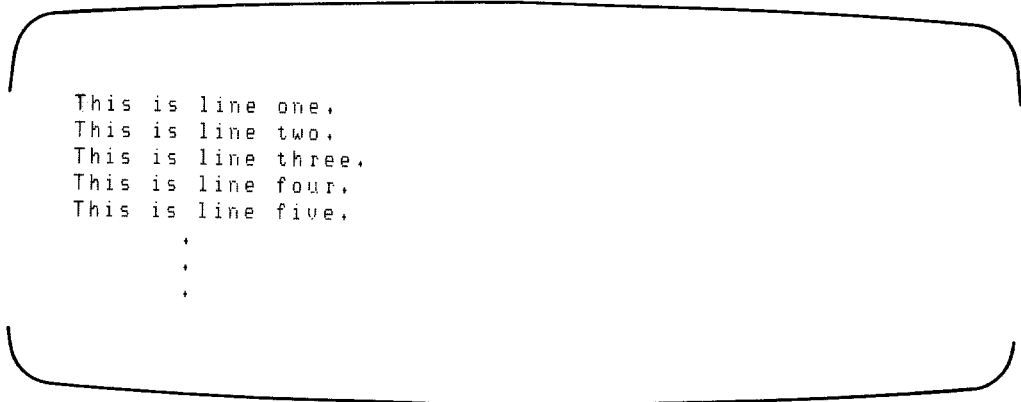
CUT

The CUT command moves the specified entities to another text buffer. The format is:

[+|-][repeat-count]CUT[+|-][entity-count][+|-]entity[=buffer]

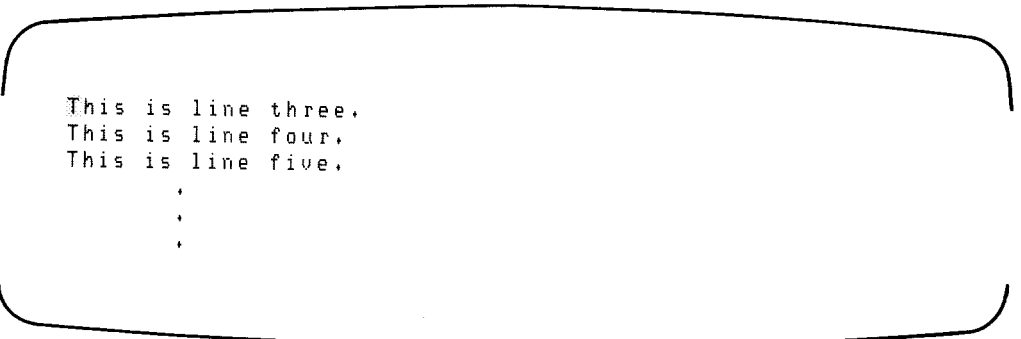
EDT deletes the moved text from the current text buffer. Use the =buffer range specification to name the receiving text buffer; EDT defaults to the paste text buffer. EDT deletes the previous contents of the receiving text buffer.

Example



```
This is line one.  
This is line two.  
This is line three.  
This is line four.  
This is line five.  
.  
.  
.
```

CUT2L=SHOW(RET)



```
This is line three.  
This is line four.  
This is line five.  
.  
.  
.
```

2APPENDL=SHOW(RET)

This is line five.

⋮

LPASTE=SHOW(RET)

This is line five.
This is line one.
This is line two.
This is line three.
This is line four.

⋮

The CUT command moves the two lines following the cursor to the text buffer you have named SHOW. EDT deletes these two lines from the current text buffer. The 2APPENDL command moves the two lines following the cursor position to the text buffer SHOW. These two lines, which are deleted from the current text buffer, are inserted at the end of the present contents of text buffer SHOW. After you move the cursor to the left margin following line five (with L), enter the PASTE = SHOW command. EDT copies the contents of the text buffer SHOW to the present cursor location.

D (Delete)

The D command deletes a specified number of entities. The format is:

[+|-][repeat-count]D[+|-][entity-count][+|-]entity[=buffer]

Example

```
This is line one.  
This is line two.  
This is line three.  
.  
.  
.
```

D2W(RET)

```
line one.  
This is line two.  
This is line three.  
.  
.  
.
```

LDL(RET)

```
line one.  
This is line three.  
.  
.  
.
```

Assume the cursor is located at the beginning of line one. To delete the first two words, you enter D2W and (RET). EDT deletes the first two words and leaves the cursor at the beginning of the third word.

To delete line two, you enter LDL and **(RET)**. EDT moves down one line in response to the first L and deletes the line in response to the DL command. The cursor is at the beginning of line three.

DEFK (Define Key)

The DEFK command defines keystrokes used in keypad editing in terms of nokeypad commands. You can redefine keys for keypad editing if you have a VT52 or VT100 terminal. EDT performs the commands that you assign with DEFK when you press the key. For more information on redefining keys, see Chapter 10.

EX (Exit)

The EX command exits EDT from nokeypad editing back to line editing. The format is:

EX

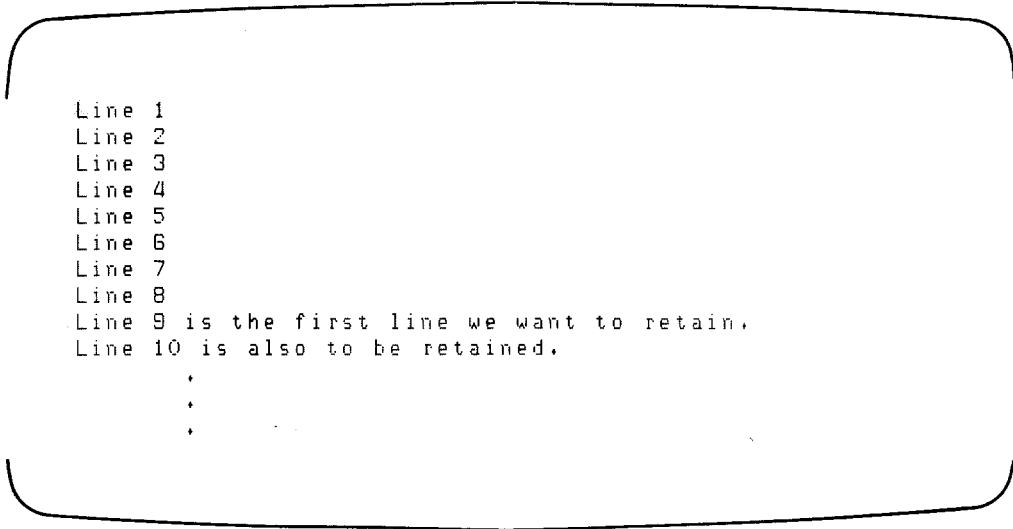
EXT (Extended)

The EXT command enters line mode commands while EDT is in change mode. The format is:

EXT

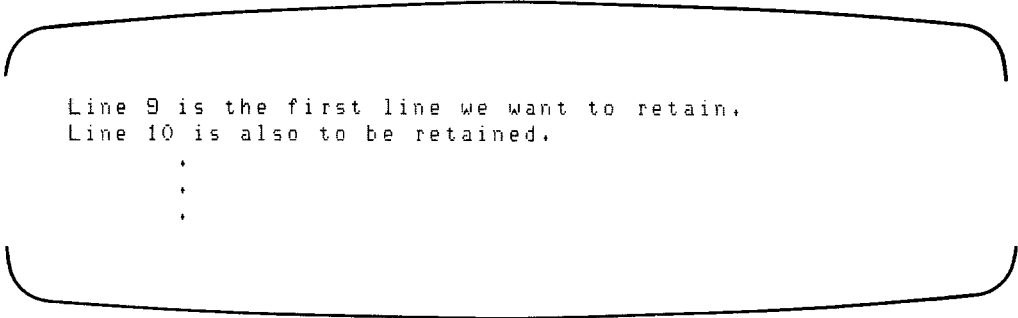
EDT accepts the rest of the line (after EXT) as a line editing command. When EDT completes the specified command, it returns you to change mode.

Example



```
Line 1
Line 2
Line 3
Line 4
Line 5
Line 6
Line 7
Line 8
Line 9 is the first line we want to retain.
Line 10 is also to be retained.
.
```

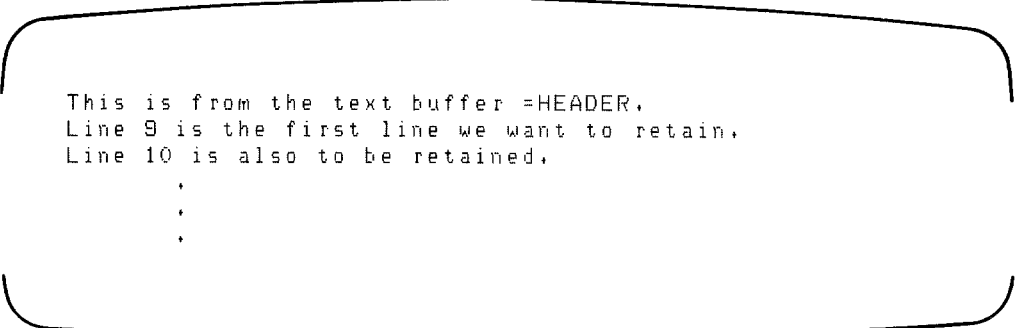
EXT DEL 1 THRU 8**(RET)**



```
Line 9 is the first line we want to retain.  
Line 10 is also to be retained.
```

```
.  
. .  
.
```

```
EXT COPY=HEADER TO 9RET
```



```
This is from the text buffer =HEADER.  
Line 9 is the first line we want to retain.  
Line 10 is also to be retained.
```

```
.  
. .  
.
```

EDT deletes all lines in the range from 1 through 8 from the current text buffer. For illustration, assume that the text buffer line numbers match the numbers shown in text. EDT then refreshes the screen, putting the line after the last line that was deleted at the top of the display. With the second EXT command, EDT copies the contents of text buffer HEADER preceding line 9 and refreshes the screen to display line 9 and one or more of the lines of the inserted text.

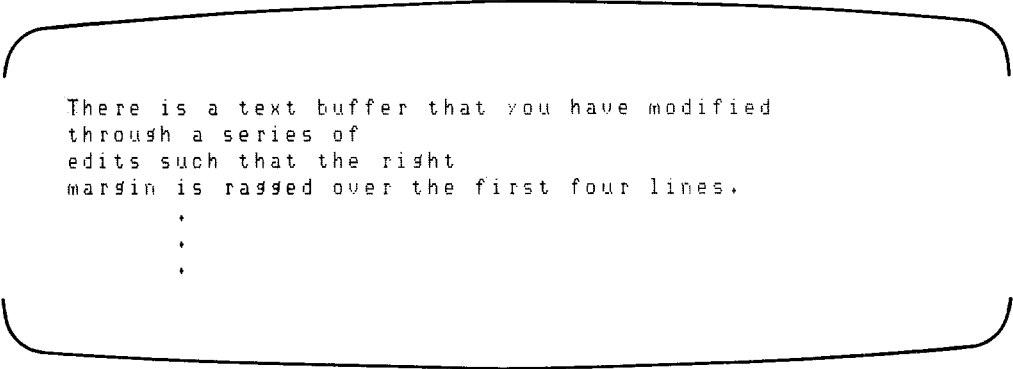
FILL

The FILL command places the maximum amount of text on each line within the bounds set by the SET WRAP command. The format is:

```
[+|-][repeat-count]FILL[+|-][entity-count][+|-]entity[=buffer]
```

The text from the succeeding lines is moved up to fill out the line. Text is broken between words; the line is filled to the word delimiter that occurs before the limit of SET WRAP is exceeded. FILL does not fill across blank lines; the lines between blank lines are filled. The default value for the fill line is 80 characters.

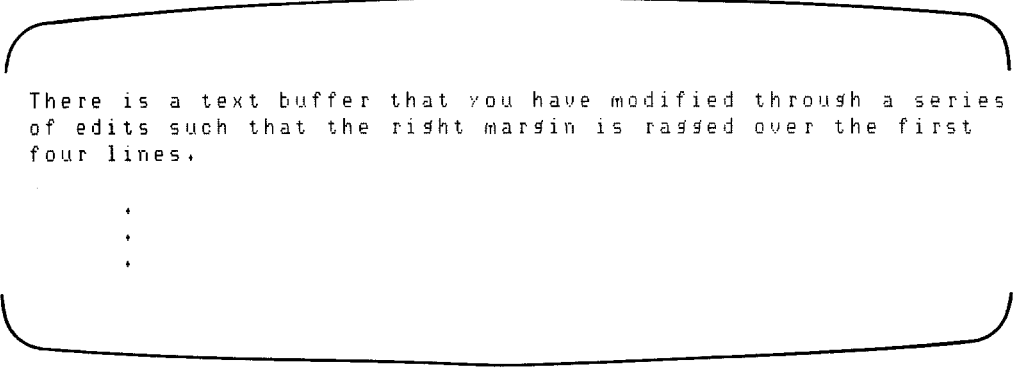
Example



```
There is a text buffer that you have modified
through a series of
edits such that the right
margin is ragged over the first four lines.
```

⋮

FILL4L(RET)



```
There is a text buffer that you have modified through a series
of edits such that the right margin is ragged over the first
four lines.
```

⋮

The first four lines represent the text buffer after a series of edits. The SET WRAP is set at 64. When you enter FILL4L, EDT fills lines, with up to 64 characters in each line, until the end of the fourth line is reached.

I (Insert)

The I command prepares the current text buffer for the insertion of text. The format is:

I

The text is inserted in front of the cursor position. EDT displays the text, at its proper location in the text buffer, as you insert it. To exit from the insert operation, press (RET).

Example

```
This is line one.  
This is line two.  
.  
.
```

I RET

```
This is line one.  
This is line two.  
This line is inserted after line two. CTRL/Z  
.  
.
```

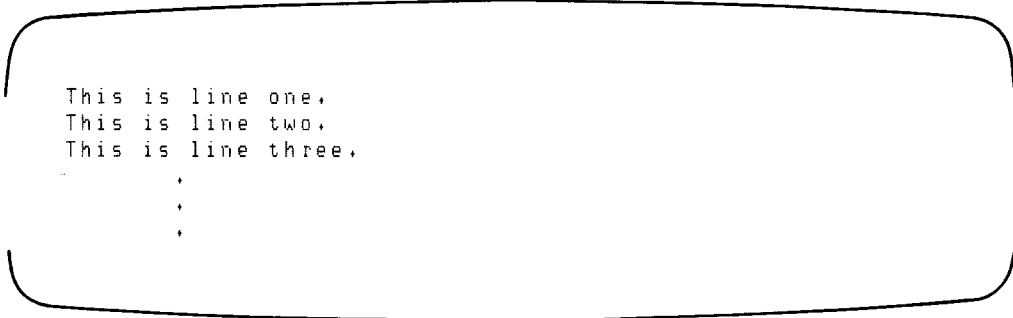
In this example the first two lines of the display are shown; the cursor is at the beginning of line 3. To insert text, enter I followed by a RET. The I appears at the bottom of the display until you exit the INSERT command. When you wish to exit the insert command, enter CTRL/Z.

Null (Implied TYPE)

The null command moves the cursor the specified number of entities. The format is:

[+|-][repeat-count][+|-][entity-count][+|-]entity[=buffer]

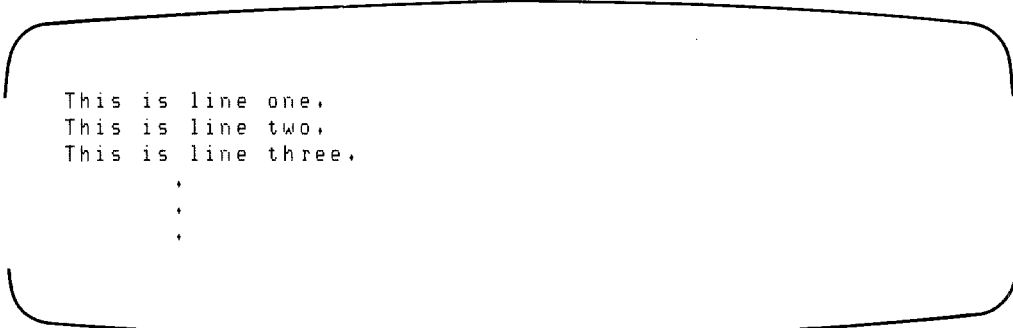
Example



```
This is line one.  
This is line two.  
This is line three.
```

.

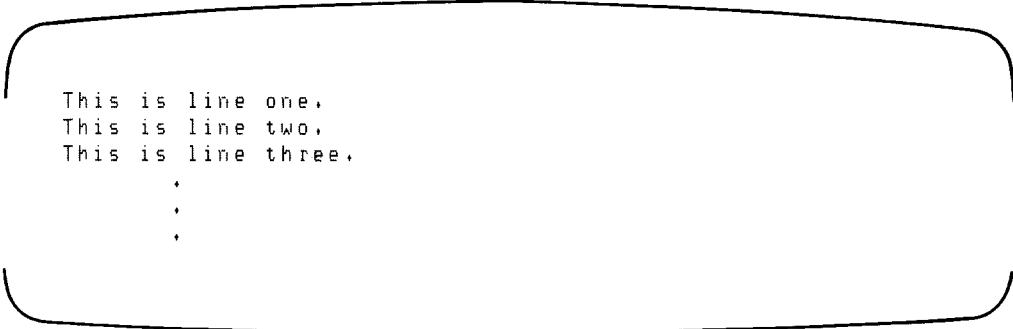
2L(RET)



```
This is line one.  
This is line two.  
This is line three.
```

.

3W3C(RET)



```
This is line one.  
This is line two.  
This is line three.
```

.

Assume the cursor is located at the first character of line one and that you want to move to the end of the third line. Enter 2L and **(RET)**; EDT moves the cursor to the start of the third line. Enter 3W3C and **(RET)** and EDT moves the cursor three words and three characters. The cursor is at the end of line three. You can perform the same sequence with one command string (2L3W3C) or with three command strings (2L, 3W, 3C). EDT executes the commands in the string in the order you entered them.

PASTE

The PASTE command copies the contents of the specified text buffer in front of the cursor location. The format is:

[+|-][repeat-count]PASTE[+|-][entity-count][+|-]entity[=buffer]

The contents of the specified buffer are unchanged by the PASTE operation. When you do not specify a text buffer, EDT defaults to the paste text buffer. See the example in the CUT command description.

QUIT

The QUIT command ends the editing session without saving your edits and returns you to the system command level prompt. The format is:

QUIT

R (Replace)

The R command deletes the text you specify and allows you to replace the deleted text. The format is:

[+|-][repeat-count]R[+|-][entity-count][+|-]entity[=buffer]

EDT deletes the selected entities and then enters insert mode. To exit from insert mode, press **(CTRL/Z)**.

Example

```
This is line one.  
This is line two.  
This is an improper entry.  
This is also an improper entry.  
This is line six.  
.  
.  
.
```

R2L(RET)

```
This is line one.  
This is line two.  
This is line three.  
This is line four.  
This is line five. CTRL/Z  
This is line six.  
.  
.  
.
```

Use the R command to replace the third and fourth lines, which are in error, with new text. Move the cursor to the first character position of line three and enter the R2L command. EDT deletes lines three and four and prepares the text buffer for the insertion of text. You exit the insert mode with CTRL/Z.

REF (Refresh)

The REF command forces EDT to refresh the entire screen. The format is:

REF

EDT normally refreshes only those portions of the screen changed by the command operations. This command is useful when the screen contains extraneous text, such as when someone sends your terminal a message via electronic mail.

S/s1/s2/ (Substitute)

The S/s1/s2/ command replaces one string of characters with another. The format is:

[+ | -][count]S/s1/s2/

Use count to define the number of substitutions and a minus sign (-) when you want a backward search. You can use any nonalphanumeric character as a delimiter, as long as the character is not used in one of the strings. For example, you could not use the slash as a delimiter to substitute the string "input/output" with "I/O". However, you could use another special character, such as:

```
S&input/output&I/O&
```


The search string (string-1) is stored in the string search buffer; the substitute string is stored in the substitute buffer.

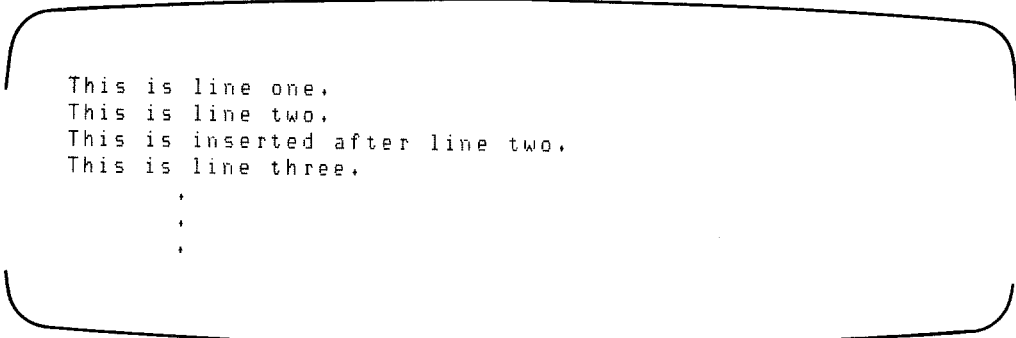
SEL (Select)

The SEL command lets you select a range of text. The format is:

SEL

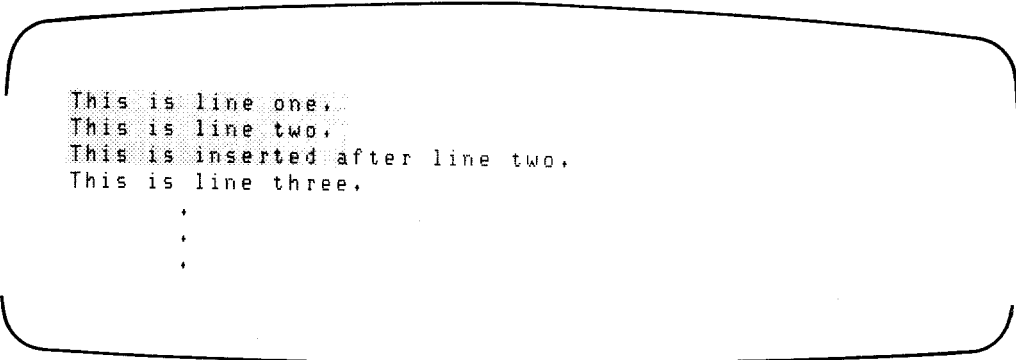
SEL marks the present cursor position. To select a range, you enter SEL at one end of the range and then move the cursor to the other end. The select range is the text between the cursor and the position marked by SEL. You can use the SEL range as an entity.

Examples



```
This is line one.  
This is line two.  
This is inserted after line two.  
This is line three.  
.  
.  
.
```

SEL(RET)
2L3W(RET)



```
This is line one.  
This is line two.  
This is inserted after line two.  
This is line three.  
.  
.  
.
```

The cursor is at the beginning of line one. By entering the SEL command, you set the beginning of line one as the start of the SEL range. The cursor marks the other end of the range; moving the cursor adjusts the other end of the range. With the cursor between "inserted" and "after", the range is from the start of line one to the beginning of "after".

SHL (Shift Left)

The SHL command shifts the screen image to the left. The format is:

[count]SHL

You control the amount of shift through your count entry, where shift equals count times 8 (8 equals one tab stop). EDT does not display the left portion (count times 8) of the text. You use the SHL command to counter the SHR command. See the example in the SHR command description.

SHR (Shift Right)

The SHR command shifts the screen image to the right. The format is:

[count]SHR

You control the amount of shift through your count entry, where shift equals count times 8 (8 equals one tab stop). You use the SHR command to counter the SHL command.

Example

```
* SET SCREEN 40(RET)
* CHANGE(RET)
```

```
The line of text is limited to 40 charact
To be able to read that which is beyond t
This is line three.
```

```
·
·
·
```

```
4SHL(RET)
```

```
characters.  
beyond the limit use the SHL command.
```

```
·  
·  
·
```

4SHR(RET)

```
The line of text is limited to 40 charact  
To be able to read that which is beyond t  
This is line three.
```

```
·  
·  
·
```

In the first part of the example, the lines are set to truncate at 40 characters. The first two lines extend beyond the 40 character limit and are truncated. The 4SHL command moves the left margin 32 characters to the left. The remainder of the first two lines can now be read. The third line is less than 32 characters long; none of the third line is visible. The SHR command restores the text to its original position.

SN (Substitute Next)

The SN command uses the s1 and s2 defined in the last substitute command (stored in the search string and substitute buffers) to replace the next occurrence of s1 with s2. The format is:

[+|-][count]SN

Use count to define the number of substitutions and a minus sign (-) when you want to search backward in the buffer.

Example

```
This is line one.  
This is line two.  
This is line three.  
.  
.  
.
```

S/line/sentence/RET

```
This is sentence one.  
This is line two.  
This is line three.  
.  
.  
.
```

2SNRET

```
This is sentence one.  
This is sentence two.  
This is sentence three.  
.  
.  
.
```

With the cursor at the start of line one, enter the S/line/sentence/ command and a RET. EDT replaces the string "line" in the first line with the string "sentence". If you then enter the command 2SN, EDT substitutes the next two occurrences of the string "line" with the string "sentence". You can achieve the same result by preceding the S command with a count of 3.

TAB

The TAB command sets the tab position in one of two ways:

1. When you have not set a tab size with SET TAB (see Chapter 9) or when the cursor is not positioned at the start of a line, the TAB command inserts a tab character at the cursor position.
2. When you have set a tab size with SET TAB and the cursor is positioned at the beginning of a line, the TAB command inserts a number of tab characters and spaces to move the cursor to the column position which is equal to the SET TAB value times the indentation level count.

The format is:

TAB

The indentation level count defines the current level of indentation in terms of the tab size. The number of columns indented is equal to the indentation level count multiplied by the tab size. The counter is set to 1 when you enter a SET TAB value. The counter is incremented by 1 with the TI command and decremented by 1 with the TD command. The counter is also set by the TC command.

Example

```
* SET TAB 4(RET)  
* SET NOKEYPAD(RET)  
* CHANGE(RET)
```

```
This is line one.  
This is line two.  
This is line three.
```

```
·  
·  
·
```

TAB␣

```
      This is line one.  
      This is line two.  
      This is line three.  
      .  
      .  
      .
```

TC␣

TI␣

You set the value for the first tab setting to 4 spaces, select nokeypad character editing, and enter the CHANGE command. Entering the TAB command moves the cursor four spaces, to character space 5. If you then move the cursor to a multiple of the SET TAB value plus one character and enter TC, the indentation level count is set to the result of dividing the value for the present cursor position by the SET TAB value (the left margin is position 0). In this example the indentation level count is set to 3. The TI command increments the indentation level count by one to the value 4.

TADJ (Tab Adjust)

The TADJ command adjusts the tab level for the selected range of lines. The format is:

[+|-][repeat-count]TADJ[+|-][entity-count][+|-]entity[=buffer]

You set the number of lines through the range specification. The tab size and repeat-count set the tab level (see TC). The tab level is adjusted by the value of repeat-count; it is incremented for a plus repeat-count and decremented for a negative repeat-count. The TADJ tab setting is the product of the SET TAB value and the indent level.

Example

```
This is the first line of text with a list following:
Item a
Item b
Item c
Item d
The list of items is complete.
.
.
.
```

TADJ4L(RET)

```
This is the first line of text with a list following:
    Item a
    Item b
    Item c
    Item d
The list of items is complete.
.
.
.
```

With the cursor at the beginning of second line, you enter TADJ4L. EDT indents the next four lines to the value of SET TAB. To increase the indent, you use the repeat-count. EDT sets the indent level to the product of repeat-count and the value of SET TAB.

TC (Tab Compute)

The TC command sets the indentation level count to the value obtained from dividing the current cursor column position by the SET TAB number. The format is:

TC

An error message occurs if the cursor is not at a multiple of the SET TAB number. EDT uses the product of indentation level count and tab size to set the first tab position for each line. Subsequent tab positions on the same line are a multiple of the terminal's tab setting (8 spaces). If you do not know the tab level, TC moves the cursor to a known column number.

TD (Tab Decrement)

The TD command decreases the indentation level count. The format is:

[count]TD

When the indentation level count is zero, the count is not changed.

TI (Tab Increment)

The TI command increases the indentation level count. The format is:

[count]TI

TOP

The TOP command places the current line at the top of the screen. The format is:

TOP

UNDC (Undelete Character)

The UNDC command inserts the contents of the character buffer into the current text buffer (just in front of the cursor). The format is:

[count]UNDC

The character buffer contains the last character deleted by one of the delete character commands. You can use count to repeat the UNDC operation; EDT inserts the contents of the character buffer into the text buffer count times.

UNDW (Undelete Word)

The UNDW command inserts the contents of the word buffer into the current text buffer (just in front of the cursor). The format is:

[count]UNDW

The word buffer contains the last word deleted by one of the delete word commands. You can use count to repeat the UNDW operation; EDT inserts the contents of the word buffer into the text buffer count times.

UNDL (Undelete Line)

The UNDL command inserts the contents of the line buffer into the current text buffer (just in front of the cursor). The format is:

[count]UNDL

The line buffer contains the last line deleted by one of the delete line commands. You can use count to repeat the UNDL operation; EDT inserts the contents of the line buffer into the text buffer count times.

^ (Circumflex)

The ^ enters a control character in your text. The format is:

[count] ^ [A . . . Z]

You enter the desired control character (A through Z) after the circumflex. EDT accepts the circumflex and alphabetic character as one character (^G, for example). Count lets you repeat the character insertion operation.

Chapter 9

The SET and SHOW Commands

SET and SHOW are line editing commands that let you specify keypad, line, and nokeypad editing functions.

The SET commands let you control the operating characteristics of EDT, such as the width of a display of text. You use the SHOW commands to display these characteristics.

The SET and SHOW commands, described in alphabetical order in this chapter, are listed in Table 9-1.

Table 9-1: SET and SHOW Parameters

| SET Command Parameters | SHOW Command Parameters |
|------------------------|-------------------------|
| — | BUFFER |
| CASE | CASE |
| CURSOR top:bottom | CURSOR |
| ENTITY | ENTITY |
| — | KEY |
| KEYPAD | — |
| LINES number | — |
| MODE | — |
| [NO]NUMBERS | — |
| [NO]QUIET | — |
| SCREEN width | SCREEN |
| SEARCH | SEARCH |
| TAB n | — |

(Continued on next page)

Table 9-1: SET and SHOW Parameters (Cont.)

| SET Command Parameters | SHOW Command Parameters |
|---|------------------------------------|
| TERMINAL [NO]TRUNCATE [NO]VERIFY — [NO]WRAP n | TERMINAL — — VERSION — |

The SET Commands

SET is a line editing command that lets you specify some of EDT's operating conditions. Once set, these conditions remain in effect until you exit the editor or until you change them with another SET command.

The following sections describe the parameters you can specify with SET.

SET CASE

When you SET CASE (UPPER or LOWER), EDT flags the selected case characters with a preceding apostrophe. The flag is displayed on the terminal, but it is not transferred to the file when you exit EDT. The format of SET CASE is:

```
SET CASE { UPPER | LOWER | NONE }
```

The EDT default for CASE is NONE, which does not flag any characters. The primary use for this qualifier is on terminals with single case capability.

Example

```
* SET CASE UPPERRET
```

With this command, EDT marks all uppercase characters with a preceding apostrophe. The display on a terminal with upper and lower case for the first part of the preceding paragraph is:

```
'When you 'S'ET 'C'A'S'E ('U'P'P'E'R or 'L'O'W'E'R) ...
```

For terminals with single case, the same copy reads:

```
'WHEN YOU 'S'E'T 'C'A'S'E ('U'P'P'E'R OR 'L'O'W'E'R) ...
```

SET CURSOR top:bottom

SET CURSOR is applicable to screen terminals and character editing only. The format is:

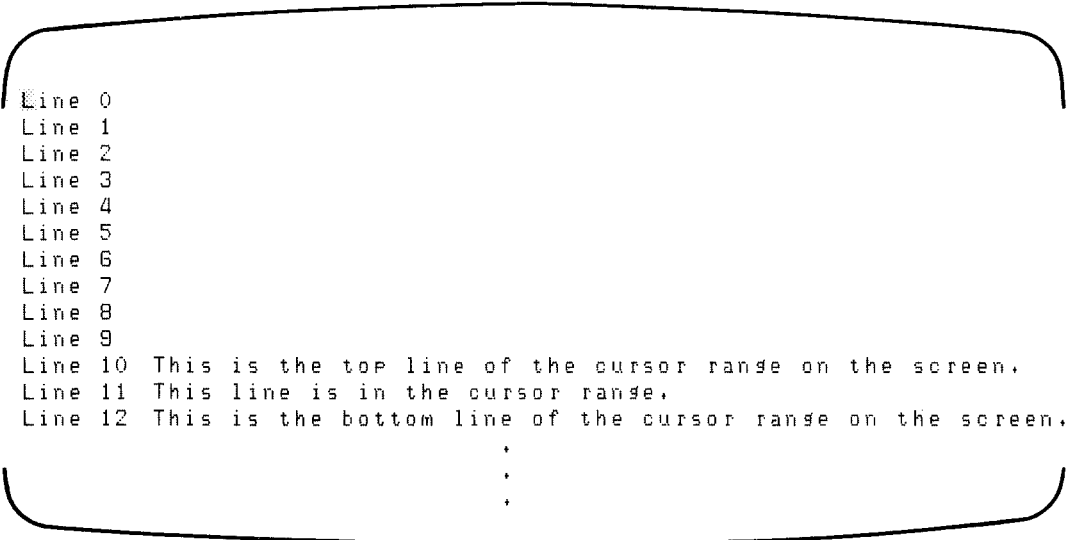
SET CURSOR top:bottom

The CURSOR top:bottom parameter sets the number of lines over which the cursor moves on the display, where top is the number of lines for the upper limit and bottom is the number of lines for the lower limit. The top and bottom counts start from the top line on the screen, which is number 0.

When you reach either limit, EDT scrolls the text so you can access the desired line(s). The allowable limits for top and bottom are 0 through 21. The default for SET CURSOR is top=7 and bottom=14. On initial text entry into a text buffer, you use the lines up to and including the value set for top.

Example

```
* SET CURSOR 10:12(RET)
* CHANGE(RET)
```



```
Line 0
Line 1
Line 2
Line 3
Line 4
Line 5
Line 6
Line 7
Line 8
Line 9
Line 10 This is the top line of the cursor range on the screen.
Line 11 This line is in the cursor range.
Line 12 This is the bottom line of the cursor range on the screen.
      .
      .
      .
```

The 11th line from the top is set as the first line of the edit area and the 13th line is the last of the edit area. When you try to move the cursor to a position outside of these lines, EDT scrolls the display up or down until the line is within the cursor range.

SET ENTITY

The SET ENTITY command sets the user-definable entities for character editing (WORD, SENTENCE, PARAGRAPH, and PAGE). Each of the entities is defined by a string of characters. The format is:

```
SET ENTITY { WORD | SENTENCE | PARAGRAPH | PAGE } 'string'
```

For the WORD and SENTENCE entities, each character in the defining string can be a delimiter. For the PAGE and PARAGRAPH entities, the entire defining string is the delimiter.

The default values for the entity delimiters are:

| | |
|-----------|------------------------------|
| Word | LF or TAB or FF or RET or SP |
| Sentence | . or ? or ! |
| Paragraph | RET RET |
| Page | FF |

Example

```
* SET ENTITY WORD ' , ' RET
* SET ENTITY SENTENCE ; RET
* CHANGE RET
```

The delimiters for WORD are space or comma. All the commands using word as an entity work up to the delimiter. When the delimiter, space, is on the right, the delimiter is included in the operation. When the delimiter, space, is on the left, the delimiter is excluded from the operation. All other word delimiters are treated as words. When a delete word command is positioned such that the first character to be deleted is a comma, EDT will delete the comma.

The delimiter for SENTENCE is the semicolon (;). All commands using a sentence as an entity work up to the delimiter. The semicolon delimiter is useful in some programming languages.

SET KEYPAD

When the KEYPAD parameter is set, the keypad controls the character editing operations. The format is:

SET { KEYPAD | NOKEYPAD }

When you set the NOKEYPAD parameter, the keyboard controls the character editing operations. The default for this parameter, for VT52 and VT100 terminals, is KEYPAD.

SET LINES number

The SET LINES number command sets the number of lines that EDT displays on your terminal for character editing. The format is:

SET LINES number

The number of lines can be set from 1 to 22. When you are editing at slow baud rates (terminal speeds), you can use this parameter to reduce the number of lines. The default for this parameter is 22.

Example

```
* SET LINES 8(RET)  
* CHANGE(RET)
```

EDT displays 8 lines at the top of the screen.

SET MODE

You can use SET MODE in a startup command file to control the editing mode which is entered at the end of the initialization process. The format is:

SET MODE { LINE | CHANGE }

SET MODE CHANGE

The SET MODE CHANGE command causes EDT to begin a session in change mode; you do not have to enter the CHANGE command.

SET MODE LINE

The SET MODE LINE command causes EDT to begin an editing session in line mode; this is the default mode.

SET [NO]NUMBERS

The SET [NO]NUMBERS command determines whether or not EDT displays the line numbers that normally appear in line editing. The format is:

```
SET { NUMBERS | NONUMBERS }
```

SET [NO]QUIET

The SET QUIET command suppresses the ringing of the bell when an error occurs in change mode. The format is:

```
SET { QUIET | NOQUIET }
```

The default for this parameter is NOQUIET.

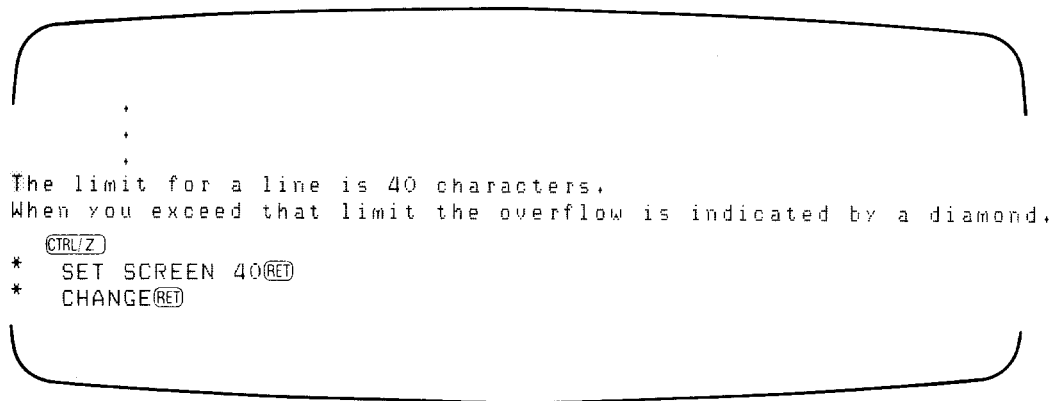
SET SCREEN width

The SET SCREEN width command controls the maximum length of a line EDT displays. The format is:

```
SET SCREEN width
```

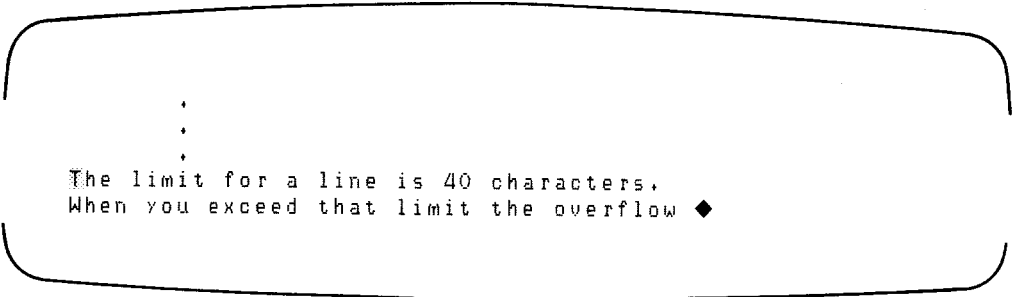
In change mode, when you insert more characters than the length set for character editing, EDT displays a solid rectangle on the VT52 and a diamond on the VT100. In line editing, EDT may display the overflow characters on succeeding lines. This depends on the terminal type. The default for this parameter is 80 characters.

Example



```

.
.
.
The limit for a line is 40 characters.
When you exceed that limit the overflow is indicated by a diamond.
*  CTRL/Z
*  SET SCREEN 40
*  CHANGE
```

The limit for a line is 40 characters.
When you exceed that limit the overflow ♦

In the first part of the example, SET SCREEN is set to the default of 80 characters and there is no overflow. You then set the maximum number of characters EDT displays on a line to 40. The first line shown in frame 2 does not exceed 40 characters. The diamond indicates that the second line exceeds 40 characters.

SET SEARCH

The SET SEARCH command sets parameters for string searching in character editing.

SET SEARCH EXACT

The SET SEARCH EXACT command causes EDT to search for exact comparisons of case in the specified string(s). The format is:

SET SEARCH { GENERAL | EXACT }

The default for this parameter is GENERAL, which disregards case.

SET SEARCH BOUNDED

The SET SEARCH BOUNDED command causes EDT to stop the search for the specified string at the next page entity marker. The format is:

SET SEARCH { BOUNDED | UNBOUNDED }

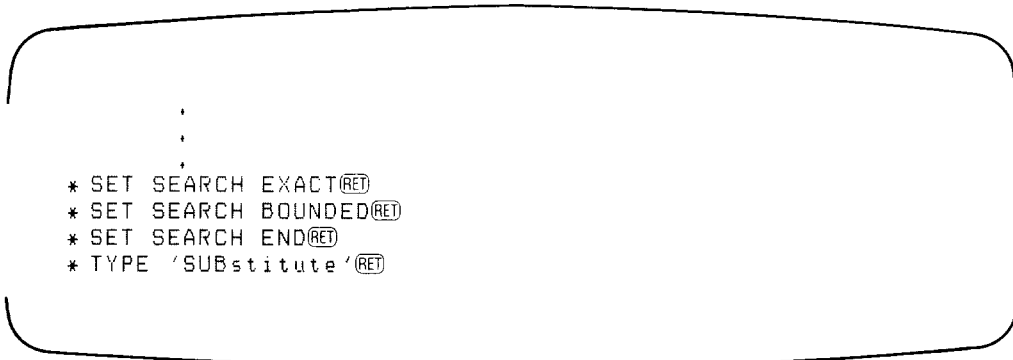
The default for this parameter is UNBOUNDED, which searches the entire buffer.

SET SEARCH END

The SET SEARCH END command causes EDT to leave the cursor at the end of the search string when it is found. The format is:

SET SEARCH { BEGIN | END }

The default for this parameter is BEGIN, which leaves the cursor at the beginning of the string. If the string is not found, the cursor position is unchanged.



```
* SET SEARCH EXACT(RET)
* SET SEARCH BOUNDED(RET)
* SET SEARCH END(RET)
* TYPE 'SUBstitute'(RET)
```

EDT searches the current text buffer, from the current cursor position to the next page marker, for the first occurrence of the string "SUBstitute." The first three characters in the string must be uppercase, and the remaining characters must be lowercase. When a string is found, EDT puts the cursor at the character position following the string.

SET TAB n

The SET TAB n command sets the number of spaces for the first tab stop in keypad character editing. The format is:

```
SET { TAB n | NOTAB }
```

The remaining tab stops for the terminal are unchanged (multiples of 8 spaces). After you have set the first tab stop with SET TAB, you can increase the first tab stop position by the value of SET TAB with (GOLD/E). You can decrement the first tab stop by the value of SET TAB with (GOLD/D).

EDT maintains an indentation level count. The count is set to one when you enter the SET TAB command. The indentation level count is incremented by one when you enter (GOLD/E) and decremented by one when you enter (GOLD/D). The first (TAB) entry for a line is the product of the value of SET TAB and the indentation level count. When the cursor is at position n times the initial SET TAB, you can set the first tab set to that position with (GOLD/A).

You can adjust the tab setting over a range of lines so that all lines in the selected range are indented. You enter the character editing SELECT function to define the start of the range. You then move the cursor to the end of the range of lines and enter (GOLD/T) preceded by a GOLD function and a + or - repeat count. The lines between the SELECT function entry and the (GOLD/T) function entry are indented by the value of SET TAB times the + or - repeat count.

Examples

The first example shows the effect of the SET TAB command on the tab settings.

```
This is the first line in the text buffer.  
This is the next line in the text buffer.  
This is the last line.
```

```
(CTRL/Z)  
* SET TAB 5(RET)  
* CHANGE(RET)
```

```
This is the first line in the text buffer.  
(TAB) This is the next line in the text buffer.  
(TAB)(TAB)(TAB) This is the last line.
```

Entering a (CTRL/Z) places EDT in line editing mode. The SET TAB command sets the first tab indent to 5 spaces. The CHANGE command returns EDT to keypad character editing. Move the cursor to the start of the second line, and press (TAB). EDT indents the line 5 spaces (the SET TAB value). Move the cursor to the start of the last line, and press (TAB) three times. EDT indents 16 spaces. The cursor is indented 5 spaces for the first tab, 3 spaces for the second tab, and 8 spaces for the third tab.

The next example shows the result of incrementing and decrementing the first tab indent by using (GOLD/E) and (GOLD/D).

```
(CTRL/Z)  
* SET TAB 12(RET)  
* CHANGE(RET)
```

```

For this example there are four lines.
This is the second line.
This is the third line.
This is the fourth line.
(TAB)      The indent from the left margin is 12 spaces.(RET)
(GOLD/E)(TAB)      The indent is now 24 spaces.(RET)
(GOLD/D)(TAB)      The indent is back to 12 spaces.(RET)

```

With the SET TAB command, set the first tab for keypad character editing to 12 spaces; this sets the indentation level count to 1. Remember the remaining tab settings for a line are the default values, multiples of eight spaces from the left hand margin. Move the cursor to the end of the current text buffer (the empty line preceding [EOB]). Indent the cursor one tab space before entering text. After adding a line to the buffer, press (GOLD/E) and (TAB). This increments the indentation level count to 2; the first tab is set to 24 (the SET TAB value times the indentation level count). After adding a line indented 24 spaces, press (GOLD/D) and (TAB). This decrements the indentation level count by one, setting the indentation for the start of the next line 12 spaces.

The next example shows how to indent a selected number of lines by using the SELECT and (GOLD/T) function keys. (See Chapter 5.)

```

(CTRL/Z)
*SET TAB 4(RET)
*CHANGE(RET)

```

```

(SELECT)
For this example there are four lines.
This is the second line.
This is the third line.
This is the fourth line.
(GOLD/T)

```

```

For this example there are four lines.
This is the second line.
This is the third line.
This is the fourth line.

```

Using the SET TAB command, set the first tab to 4 spaces and enter the CHANGE command. (The text is from the example above.) EDT displays the contents of the text buffer. Move the cursor to one end of the range of lines you wish to indent, and press the SELECT function key. Move the cursor to the other end of the range, and press `(GOLD/T)`. EDT indents the selected range of lines four spaces: the remainder of the text buffer is unchanged. A `(GOLD/3)` followed by `(GOLD/T)` would shift the selected lines 12 spaces (3 times the SET TAB of 4).

The next example shows the use of the `(GOLD/A)` function key to set the indentation level count and subsequent first tab indents.

```

(CTRL/Z)
* SET TAB 5(RET)
* CHANGE(RET)

```

```

For this example there are four lines.
This is the second line.
This is the third line.
This is the fourth line.
(Move cursor to 25 spaces) (GOLD A)
(TAB)                      The first tab stop is now 25 spaces.(RET)
(GOLD D)(TAB)              The first tab stop is now 20 spaces.
[EOB]

```

Assume that the text buffer contains the four lines of text. The `CTRL/Z` places EDT in line editing mode. Using the SET TAB command, set the first tab to 5 spaces (the indentation level count is set to 1), and enter the CHANGE command. Move the cursor to the line following the text, and indent the cursor 25 spaces (a multiple of the SET TAB value). Enter a `GOLD/A`; this sets the first tab to 25 spaces and sets the indentation level count to 5 ($25/5=5$). You can decrement or increment the tab level counter by pressing the `GOLD/D` and `GOLD/E` function keys respectively (each increment is 5 spaces).

SET TERMINAL

The SET TERMINAL command is used to identify the type of terminal in use. The format is:

```
SET TERMINAL { HCPY | VT52 | VT100 }
```

EDT gets the terminal type from the operating system. You can override the terminal type with the SET TERMINAL command. You can set the terminal to VT100, VT52, or HCPY.

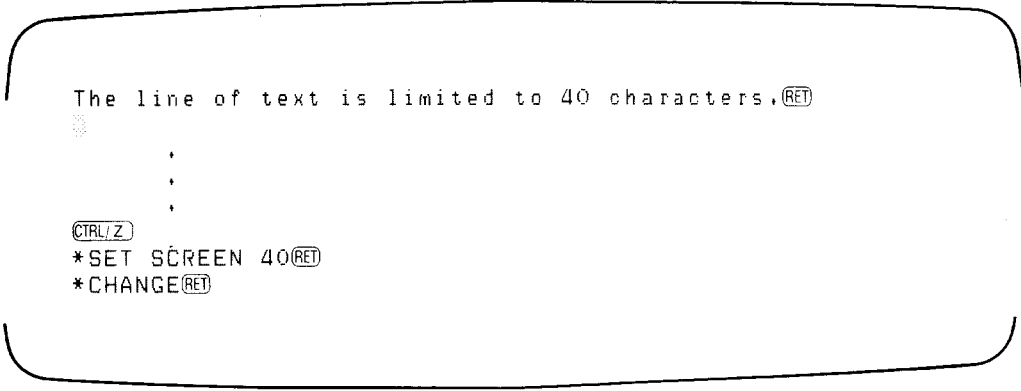
SET [NO]TRUNCATE

The SET TRUNCATE command ends the display of a line at the value of SET SCREEN. The format is:

```
SET { TRUNCATE | NOTRUNCATE }
```

The default is SET TRUNCATE. When you use SET NOTRUNCATE, the input text which exceeds the SET SCREEN value is displayed on succeeding lines. You use SET NOTRUNCATE when you want to display lines that exceed the screen width.

Example



```
The line of text is limited to 40 characters.(RET)
.
.
.
CTRL/Z
*SET SCREEN 40(RET)
*CHANGE(RET)
```

```
The line of text is limited to 40 chara ♦
```

```
·  
·  
·
```

```
CTRL Z
```

```
* SET NOTRUNCATE RET  
* CHANGE RET
```

```
The line of text is limited to 40 chara  
♦ cters.
```

```
·  
·  
·
```

In the first part of the example, the screen display limit is set to 40 characters. The `CHANGE` command places EDT into character editing. The input line exceeds 40 characters; therefore, EDT truncates the line to the limit set by `SET SCREEN`. The diamond indicates that there are additional characters in the line that EDT does not display.

The `CTRL Z` places EDT in command level. Enter `SET NOTRUNCATE` and the `CHANGE` command. The input line again exceeds the 40 characters of `SET SCREEN`, but this time the excess is displayed on the next line. The diamond indicates that this line is an overflow from the preceding line.

SET [NO]VERIFY

When you use the `SET VERIFY` command, the commands from command files and macro commands are printed when they are executed. The format is:

```
SET { VERIFY | NOVERIFY }
```

The default for this parameter is `NOVERIFY` — the command files and the macros are not printed when executed.

SET [NO]WRAP n

The SET WRAP n command sets a line length limit of n character positions. The format of this command is:

```
SET { WRAP n | NOWRAP }
```

The SET WRAP command is used in two ways:

1. When you are inserting text in character editing and the cursor position exceeds the value of n, EDT tries to wrap a full word to the next line. The word that exceeds the limit of n characters is moved to the following line. If there are no spaces in the character string, the line can be any length up to 255 characters. When you insert text in the middle of a line, the line can extend beyond the right hand margin without wrapping.
2. When you use the FILL command, SET WRAP sets the right hand margin for the fill. EDT fills each line in the range to the word delimiter nearest the limit n.

The default for this parameter is NOWRAP.

Example

```
* SET WRAP 40(RET)
* CHANGE(RET)
```

```
This line contains 40 characters of text
This line contains more than 40
characters of text.␣
```

EDT sets the right margin to 40 characters. The first line of input contains the limit of characters. If you placed a period at the end of the first sentence (after "text"), EDT would write the word "text" and the period on the next line. The next line exceeds the limit of 40 characters within the word "characters," so EDT moves "characters" to start a new line.

The SHOW Commands

SHOW is a line editing command that displays selected information on the current state of EDT, including the information you specify with the SET commands. The following sections describe the parameters you can specify with SHOW.

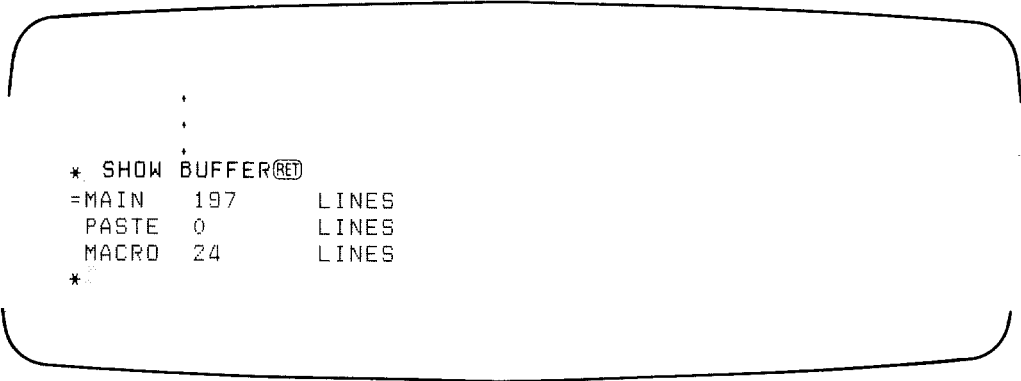
SHOW BUFFER

The SHOW BUFFER command lists the buffers in use during the current editing session and the number of lines of text in each buffer. The format is:

SHOW BUFFER

The MAIN and PASTE buffers are always displayed under the SHOW BUFFER command even when they are empty. EDT marks the name of the current text buffer with an equal sign (=). If the line count is followed by an asterisk, there are more lines in the input file which have not been read by EDT; thus, the number of lines listed is not always exact.

Examples



```
* SHOW BUFFERRET
=MAIN 197 LINES
PASTE 0 LINES
MACRO 24 LINES
*
```

EDT displays the names of all text buffers used in this editing session. MAIN is the current text buffer.

SHOW CASE

The SHOW CASE command shows the current case setting (upper, lower, or none). The format is:

SHOW CASE

CASE is defined in the SET CASE command.

SHOW CURSOR

The SHOW CURSOR command shows the current cursor range. The format is;

SHOW CURSOR

CURSOR is defined in the SET CURSOR command.

SHOW ENTITY

The SHOW ENTITY command shows the current setting for the specified entity (WORD, SENTENCE, PARAGRAPH, and PAGE). The format is:

SHOW ENTITY { WORD | SENTENCE | PAGE | PARAGRAPH }

The entities are defined in the SET ENTITY command.

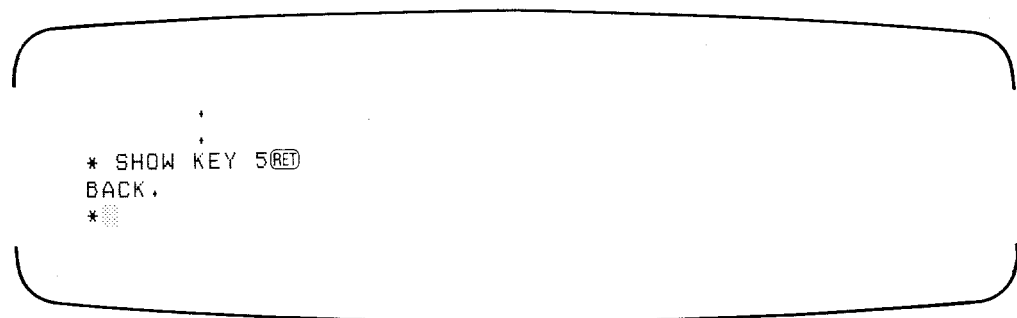
SHOW KEY

The SHOW KEY command shows the definition of the specified key in change mode. The format is:

SHOW KEY { [GOLD] { number | CONTROL letter } | GOLD character }

You can change the definition of the key with the DEFINE KEY command.

Example



The example assumes that EDT is in line editing. Enter the SHOW KEY command and the key (the numbers for the keypad keys are shown in Figure 10-1). EDT responds by displaying the current command value for that key.

SHOW SCREEN

The SHOW SCREEN command shows the current setting for the maximum length of a line EDT displays. The format is:

SHOW SCREEN

SCREEN is defined in the SET SCREEN command.

SHOW SEARCH

The SHOW SEARCH command shows the current search parameters. The format is:

SHOW SEARCH

SEARCH is defined in the SET SEARCH command.

SHOW TERMINAL

The SHOW TERMINAL command shows your terminal type (Hard-copy, VT52, or VT100). The format is:

SHOW TERMINAL

SHOW VERSION

The SHOW VERSION command shows the version number of EDT. The format is:

SHOW VERSION

Chapter 10

Redefining Keys

All of the keypad keys and several keyboard and CONTROL keys are assigned specific functions in EDT. However, you can use these redefined functions only during keypad editing. This chapter describes how to redefine the functions of these keys.

Key Concepts

There are three things you need to know about keypad editing functions in order to redefine keys:

1. Keypad editing functions are based on nokeypad commands. When you redefine a key, you are really assigning one or more nokeypad commands to the key.
2. You can redefine all keypad keys (except GOLD), and several of the keyboard and CONTROL keys.
3. There are three ways you can redefine keys:
 - In line editing, use the DEFINE KEY command
 - In keypad editing, use `CTRL/K`
 - In nokeypad editing, use the DEFK command

Each of the keypad and CONTROL keys has a string of characters associated with it that defines the functions of the key. The string consists of nokeypad editing commands. For example, the BACKUP function on the keypad is described by the string BACK., which is a nokeypad command.

When you define keys, the following rules apply:

1. A string can contain multiple commands.
2. A string ending in a period is executed immediately when you press the key associated with that string.
3. If a string does not end in a period, EDT buffers it (holds it for execution) until you enter a command that does end in a period.
4. A string can contain a question mark. The ? followed by a quoted string causes the string to prompt you for input. The string that follows the ? must be enclosed in single or double quotes.

For example, the string “?“Search for:.” causes the following prompt to appear on the screen.

```
Search for:
```

A user can enter up to 64 characters in response to the prompt.

DEFINE KEY Command

DEFINE KEY is a line editing command that lets you redefine keypad keys in terms of nokeypad commands.

The format of the DEFINE KEY command is:

```
DEFINE KEY {[GOLD] { number | CONTROL letter } | GOLD character } AS 'string'
```

In the DEFINE KEY format, you must choose one of the options between braces ({ }); the OR symbol (|) separates your choices. Brackets ([]) show where you can use GOLD to specify the alternate function of a keypad key or a control key. The terms shown in the format line are as follows:

- *number* refers to the number of the selected keypad key (see Figure 10-1).
- *CONTROL letter* means that you type CONTROL and then a keyboard character from A to Z. “Letter” can be either uppercase or lowercase; EDT does not distinguish between the two when you redefine keys.
- *GOLD* refers to the GOLD keypad key, which lets you access the alternate functions of the keypad keys or of the control keys.
- *GOLD character* means that you type GOLD and any of the keyboard keys except 0-9, !, %, ', and ". You can specify the DELETE key by using the reserved word DELETE.
- *'string'* refers to one or more nokeypad commands that you use to redefine the key.

Figure 10-1: Keypad Numbers

| | | | |
|-------------|-----------|-----------|-----------|
| GOLD | 10 | 11 | 12 |
| 7 | 8 | 9 | 13 |
| 4 | 5 | 6 | 14 |
| 1 | 2 | 3 | 15 |
| 0 | | 16 | 21 |

A. VT52 Keypad

| | | | |
|-----------|-----------|-----------|-----------|
| 12 | 13 | 15 | 14 |
|-----------|-----------|-----------|-----------|

| | | | |
|-------------|-----------|-----------|-----------|
| GOLD | 10 | 11 | 17 |
| 7 | 8 | 9 | 18 |
| 4 | 5 | 6 | 19 |
| 1 | 2 | 3 | 21 |
| 0 | | 16 | |

B. VT100 Keypad

For example, to define the standard (non-GOLD) function of a keypad key, enter the key number. To define the alternate function for a keypad key, enter GOLD and the key number.

The following example uses the DEFINE KEY command to define a string without an ending period.

Enter the following text and move the cursor to the "w" in the word "well". Then press **CTRL/Z** to start line editing:

```
Littleton is well known for(RET)
its health care facilities;(RET)
it boasts a top-rated ambulance(RET)
service and two nearby hospitals.(RET)

[EOB]
```

CTRL/Z

Use the DEFINE KEY command to assign the DELETE function to keypad key 7. Since you do not use an ending period, EDT will not execute the command until you assign an entity:

```
*DEFINE KEY 7 AS "+D"(RET)
*CHANGE(RET)
```

Press keypad key number 7, and then press the WORD function key to delete one word:

D + **WORD**
COMMAND **CHNGCASE**

```
Littleton is known for
its health care facilities;
it boasts a top-rated ambulance
service and two nearby hospitals.

[EOB]
```


You can also use the redefined key with another entity, such as LINE:



it boasts a top-rated ambulance
service and two nearby hospitals.

In the following example, you redefine the alternate function of keypad key 1 as the ? character, followed by the string 'Enter nokeypad subcommands'. When you use the redefined key, the quoted string is displayed at the bottom of the screen as a prompt. Your message can be a reminder of the special function you have assigned to the key.

Press **CTRL/Z** to return to line editing. Then type the following after the asterisk:

```
*DEFINE KEY GOLD 1 AS "?"'Enter nokeypad subcommands:'. "(RET)
*CHANGE(RET)
```

Press GOLD and keypad key 1 to see the results:

Enter nokeypad subcommands:

The entire command consists of the string that is typed in response to the "Enter nokeypad subcommands:" prompt. This allows you to enter a nokeypad command string during keypad editing.

CTRL/K

In keypad editing, you define keys with **CTRL/K**. When you press **CTRL/K**, EDT displays the message:

Press the key you wish to define

You select the key to redefine by pressing it. EDT responds with:

Now enter the definition terminated by ENTER

Enter the function definition either by using nokeypad editing commands or by pressing keypad function keys. Your entries for the function appear at the bottom of the screen. Press the ENTER function key when your command string is complete. Use the SHOW KEY command to display the definition of the key.

Use **DEL** to edit errors in the commands you type. You can use **CTRL/U** to cancel the DEFINE KEY sequence. However, you cannot cancel the command once you press the ENTER function key to execute it.

You can use control characters in your key definition. When you press a control key, the circumflex and control characters are inserted into the command string. For example, **CTRL/Z** is entered as **^Z**.

When you press certain characters, such as **RET** and **LF**, EDT inserts the circumflex character combination. For example, **RET** generates **^M**, and **LF** generates **^J**.

To use keypad keys such as **→** in a keypad definition, press the key.

Examples

Assume you are not a very good typist and frequently transpose characters. To fix these errors manually, you delete the first character, move forward one character, and retype the deleted character. After making another typing error, you decide to use the DEFINE KEY function to make the correction operation automatic.

This sentence has a trasnposed letter.

CTRL/K

Press the keypad key to be defined:

GOLD + **5**

Then enter your definition, ending it with ENTER:

DEL C
UND C + **→** + **GOLD** + **DEL C**
UND C + **.** + **ENTER**
SUBS

When you press **CTRL/K**, EDT prompts you with “Press the key you wish to define”. You press the keypad key **GOLD/5**, and EDT prompts you with “Now enter the definition terminated by ENTER”. When you press keys to define **GOLD/5**, the nokeypad editing values for those keys appear at the bottom of the screen. The finished string is D+C+CUNDC., which disappears from the screen when you press ENTER.

Move the cursor to the “s” in “trasnposed” and press **GOLD/5**. The “s” and the “n” are transposed.

The next example shows how you use a defined key to insert a frequently used word or phrase.

CTRL/K

Press the key you wish to define

GOLD + **0**

Now enter the definition terminated by ENTER

Ienvironment **CTRL/Z** + **•** + **ENTER
SUBS**

The finished string for your entry is Ienviroment [^]Z., where **CTRL/Z** is displayed as [^]Z.

When you want to insert “environment”, press **GOLD/0**.

The next example shows how to define two keys to do multiple string substitutions in keypad character editing. This definition uses the ? to prompt you for both the search string and the substitute string. You define one key to accept the search and substitute strings and perform the first substitution and another key to substitute for the next occurrence of the search string.

CTRL K

Press the key you wish to define

CTRL F

Now enter the definition terminated by ENTER

S/?'SUBS: '/' WITH: '/'

ENTER
SUBS

CTRL K

Press the key you wish to define

CTRL R

Now enter the definition terminated by ENTER

SN" " .

ENTER
SUBS

The first definition is a substitute command, and EDT prompts you for the two strings when you press **CTRL/F**. The pair of double quotes at the end of the command string causes EDT to search for the next occurrence of the string. If you enter a slash in either string, the command is invalid; the string is delimited by the slash. To avoid this, replace the slashes in the definition with a character you are not likely to use in the strings, such as a control character.

The second definition is a substitute next command; it replaces the next occurrence of the string and searches for another one.

Using a Repeat Count

If you define a key as a series of commands and wish to use a repeat count with the key, enclose the series of commands in parentheses. EDT builds a command from your input and interprets your input when you press the key. For example:

```
+L I*^Z.
```

The function defined for this key is 'move to the beginning of the next line and insert an asterisk.' Now suppose you used a repeat count of 10 to insert asterisks at the beginning of the next 10 lines. The command EDT sees after you press the keys is:

```
10+L I*^Z
```

The repeat count of 10 applies only to the +L command, so EDT moves forward 10 lines and then inserts the asterisk. This is not what you had in mind. Now change the definition to:

```
10(+L I*^Z)
```

The repeat count now appears in front of the left parentheses and the entire string inside the parentheses is executed ten times. This has the desired effect of putting an asterisk at the front of each of the next ten lines.

Appendix A

Error Messages

EDT contains a set of error messages that identify problems and assist you in the completion of the present editing operation.

Most error messages consist of a pointer line, which is a circumflex character (^), and the error message. The pointer indicates the position of the error in the command.

The error messages, arranged alphabetically, and their explanations follow.

'.' required

The command requires a period (.) or other special character shown enclosed in single quote marks.

Aborted by CTRL/C

This message occurs when you are in keypad change mode and enter a **CTRL/C** after entering a command.

Advance past bottom of buffer

The command indicates that you attempted to move the cursor past the end of the buffer ([EOB]).

Attempt to CUT or APPEND to current buffer

This message occurs when you have tried to perform a CUT or APPEND and have named the current text buffer as the destination text buffer. The default destination is the PASTE text buffer.

Attempt to PASTE the current buffer

The PASTE command attempted to paste text into the same text buffer containing the text to be pasted.

Backup past top of buffer

The command entered would move the cursor to a position preceding the first line (top) of the buffer.

Change mode may be entered only from the terminal

You cannot enter change mode when you run EDT from a batch command file or from your startup command file. Because there is no terminal associated with this job, you cannot enter change mode.

Command buffer exhausted

The string of change mode commands exceeds 255 characters.

Command file could not be opened

The command file given in the command line cannot be opened. EDT will also display an associated file system message for the error.

Command file does not exist

The command-file parameter in the command line does not exist in the specified directory.

Consistency check failed, please check your file

There is a discrepancy between the number of lines and characters entered during the editing session and the number of lines and characters present when the editing session ends. You should check for possible errors in your output file. This message indicates that there is a problem in EDT.

Could not align tabs with cursor

The cursor is at a position where it is not evenly divisible by the tab size when the tab compute function was used.

Destination for MOVE or COPY not found

The range specification in the command does not exist.

Entity must be WORD, SENTENCE, PAGE, or PARAGRAPH

The SET ENTITY command must include one of the four entity options listed.

Error in command

The command entered is invalid.

Error in command option

The command includes a /name where name is not a valid option.

Error in range specification

The command requires a range specification which must be complete.

File specification required

A file specification is required as a part of the command (WRITE, PRINT, or INCLUDE).

Help file could not be opened

The requested help file cannot be accessed. This indicates an error in system storage (for example, improper installation of the operating system).

Input file could not be opened

The command contains faulty syntax, or you specified a nonexistent directory.

Input file does not exist

The input file is not contained in the specified directory.

Input record too large, truncated to 255 characters

A record in the input file exceeds 255 characters.

Insufficient memory

There is insufficient memory to complete the last command. This message can occur when you define a new text buffer or use the DEFINE KEY command.

Invalid buffer name

You have used improper syntax for the buffer name in the command.

Invalid entity

The entity portion of the change mode command is not recognized.

Invalid option for that command

You have used an /OPTION where it is not allowed for that command.

Invalid parameter for SET or SHOW

The SET or SHOW command does not use one of the listed parameters (see Chapter 8).

Invalid subcommand

You have used an improper name for the change mode command.

Invalid value in SET command

The command has an invalid keyword.

I/O error on work file

EDT is unable to access its text storage area for the file. An additional message will explain the error further, such as "Device full" or "Device write locked".

Journal file could not be opened

The journal file is not within your defined privilege.

Line exceeded 255 characters, truncated

The input for the line exceeds 255 characters; the excess is deleted.

MACRO or KEY required

The DEFINE command is incomplete. You must include either MACRO or KEY in the command.

No definition

You have requested a SHOW KEY definition for an undefined key.

No output file name

You have used the EXIT command without having specified a file name, either in the EXIT command or in the command line.

No select range active

You did not create a select range prior to entering the APPEND or CUT command.

No such line

There are no original line numbers for the specified range.

Numeric value required

The command must have a numeric value at the point of the ^ in the command.

Output file could not be opened

EDT will display another message describing the error.

Parenthesis mismatch

The number of right hand parentheses `[])]` does not match the number of left hand parentheses `[([`. This error occurs when you are entering a change mode command string.

Parsing stack overflow

The command has caused the memory space for the parse data to be filled before the command could be validated. Check your command string. If it is valid, reenter the command in segments.

Please answer Y(es), N(o), Q(uit), or A(ll)

This is the prompt that occurs when you have selected the query qualifier and have failed to answer with one of the above in response to the ? prompt.

Quoted string required

The command requires a quoted string. The ^ indicates the position of the required quoted string.

Range must be contiguous

The range specification for the RESEQUENCE command must be contiguous lines.

Range specified by /SEQUENCE would cause duplicate or non-sequential numbers

You have range specifications in the RESEQUENCE command that would cause duplicate or non-sequential numbers.

String was not found

The string defined in the range specification cannot be found.

That key is not definable

The key selected for the DEFINE KEY command is not available for definition.

Unexpected characters after end of command

The command contains a string of one or more characters at the end of the command which are not part of the command. The rest of the command is valid.

Unrecognized command

EDT does not recognize or support the command entered. Most likely you have incorrectly specified the command.

Unrecognized command option

The command includes an invalid option/qualifier.

Work file overflow

You have exceeded 65536 blocks of text in this editing session.

Working

This message is displayed when the command operation requires more than a minimum amount of time to complete. It tells you that EDT is responding to your command.

Appendix B

ASCII Decimal Equivalents

The following table lists the decimal equivalents of ASCII characters.

| Decimal Value | ASCII Character | Decimal Value | ASCII Character | Decimal Value | ASCII Character |
|---------------|-----------------|---------------|-----------------|---------------|-----------------|
| 0 | NUL | 27 | ESC | 54 | 6 |
| 1 | SOH | 28 | FS | 55 | 7 |
| 2 | STX | 29 | GS | 56 | 8 |
| 3 | ETX | 30 | RS | 57 | 9 |
| 4 | EOT | 31 | US | 58 | : |
| 5 | ENQ | 32 | SP | 59 | ; |
| 6 | ACK | 33 | ! | 60 | < |
| 7 | BEL | 34 | " | 61 | = |
| 8 | BS | 35 | # | 62 | > |
| 9 | HT | 36 | \$ | 63 | ? |
| 10 | LF | 37 | % | 64 | @ |
| 11 | VT | 38 | & | 65 | A |
| 12 | FF | 39 | " | 66 | B |
| 13 | CR | 40 | (| 67 | C |
| 14 | SO | 41 |) | 68 | D |
| 15 | SI | 42 | * | 69 | E |
| 16 | DLE | 43 | + | 70 | F |
| 17 | DC1 | 44 | , | 71 | G |
| 18 | DC2 | 45 | - | 72 | H |
| 19 | DC3 | 46 | . | 73 | I |
| 20 | DC4 | 47 | / | 74 | J |
| 21 | NAK | 48 | 0 | 75 | K |
| 22 | SYN | 49 | 1 | 76 | L |
| 23 | ETB | 50 | 2 | 77 | M |
| 24 | CAN | 51 | 3 | 78 | N |
| 25 | EM | 52 | 4 | 79 | O |
| 26 | SUB | 53 | 5 | 80 | P |

(Continued on next page)

ASCII Decimal Equivalents (Cont.)

| Decimal Value | ASCII Character | Decimal Value | ASCII Character | Decimal Value | ASCII Character |
|---------------|-----------------|---------------|-----------------|---------------|-----------------|
| 81 | Q | 97 | a | 113 | q |
| 82 | R | 98 | b | 114 | r |
| 83 | S | 99 | c | 115 | s |
| 84 | T | 100 | d | 116 | t |
| 85 | U | 101 | e | 117 | u |
| 86 | V | 102 | f | 118 | v |
| 87 | W | 103 | g | 119 | w |
| 88 | X | 104 | h | 120 | x |
| 89 | Y | 105 | i | 121 | y |
| 90 | Z | 106 | j | 122 | z |
| 91 | [| 107 | k | 123 | { |
| 92 | \ | 108 | l | 124 | Vertical Line |
| 93 |] | 109 | m | 125 | } |
| 94 | ^ or ↑ | 110 | n | 126 | ~ Tilde |
| 95 | — or ← | 111 | o | 127 | DEL RUBOUT |
| 96 | ` Grave Accent | 112 | p | | |

Appendix C

Terminals with Low Baud Rates

Screen editing is designed to display a large number of characters on the screen at high speeds. For example, keypad editing works efficiently at 9600 *baud*, which is 960 characters per second. However, there are speeds at which EDT is unable to display characters as quickly as you type them.

If your terminal is connected to a computer system by a telephone line, your terminal probably operates at a speed close to 300 baud (30 characters per second). 300 baud is rather slow for keypad editing. You may prefer to use nokeypad or line editing at such slow speeds.

It is possible for you merely to accept the slow response time of keypad editing at low baud rates. Avoiding the insertion of text in the middle of lines could help. However, there are steps you can take to improve keypad editing response time.

One way to improve response time is to limit the number of lines that appear on the screen as you edit. It is important to restrict cursor movement to this number of lines. (You can set lines from 1 to 22, and set cursor movement from lines 0 to 21.)

For example, the following line editing commands set the number of lines to 8 and permit cursor movement within this range of lines:

```
*SET LINES 8  
*SET CURSOR 0:7
```

You can change these restrictions during keypad editing with the **COMMAND** function.

Glossary

alphanumeric

A contraction of alphabetic-numeric; the set of characters that compose text; characters include letters, numerals, and exclude special characters.

ASCII

The acronym for American Standard Code for Information Interchange; a standardized code representing the 128 characters in which text is recorded. The decimal values for the ASCII characters are contained in Appendix B.

buffer

A temporary storage area used to contain text.

case

The state of the alphabetic characters. Capital letters are upper case.

character

A symbol representing an ASCII code. See also alphanumeric.

character buffer

A temporary storage area used to store the last character deleted by an EDT delete character operation.

character editing

Editing text at the character level; an operational mode of EDT, entered through the CHANGE command.

character string

A sequence or group of connected characters.

command

An instruction typed by the user at a terminal or included in a command file which requests the operating system to perform some well-defined procedure.

command file

A file containing command strings.

command line

The command for entering EDT.

command string

A line (or set of continued lines), normally terminated by typing a `RET`, containing a command and (optionally) information modifying the command. The fullest form of the command string contains a command, its qualifiers, and its parameters (file specifications, for example) and their qualifiers.

contiguous

Contiguous lines or characters are those which are adjacent to one another.

control key

The keyboard character that causes a control action; a control key is usually the combination of the CTRL key and an alphabetic key.

CTRL/A or GOLD/A

Sets the tab position to the present cursor position.

CTRL/C

Ends the current keypad operation and returns you to the keypad command level.

CTRL/D or GOLD/D

In keypad character editing, decreases the tab indentation level count by one.

CTRL/E or GOLD/E

In keypad character editing, increases the tab indentation level count by one.

CTRL/I

Duplicates the function of the TAB key.

CTRL/K

In keypad character editing, lets you use the DEFINE KEY function.

CTRL/Q

Restarts terminal output that was suspended by **CTRL/S** .

CTRL/S

Suspends the terminal output until you press **CTRL/Q** .

CTRL/T or GOLD/T

In keypad character editing, sets the tab indent over a range of lines.

CTRL/U or GOLD/U

In keypad character editing, deletes the text from the cursor position to the left margin.

CTRL/W

In keypad character editing, refreshes the screen display.

CTRL/Z

Exits you from insert mode or keypad character editing and returns you to line editing.

decimal number

A number in the numbering system with a base of 10; a numbering system composed of 10 possible symbols (0 through 9) with each number position representing that symbol times some power of 10.

default

The act of omitting information in a computer operation so that the computer makes a predefined assumption about the operation; an assumption made by the operating system when no specific value is provided by the user.

delimiter

A character that limits a string of characters and therefore cannot be a member of the string.

directory

A file, used to locate files on a mass storage device, that contains a list of complete file names and their unique internal identifications.

direction

Direction in EDT can be forward or backward; the symbol for forward direction is the plus sign (+) and the symbol for backward direction is the minus sign (−).

editing session

The interval between entering and exiting EDT.

entity

Units of text from 1 to n characters on which EDT commands operate; for example, character, word, line.

error message

A message displayed by EDT indicating that EDT cannot perform the operation you requested.

exit

The means of halting a computer cycle of operation (EDT); the ending of an editing session. When capitalized in this manual, EXIT is a command.

field

A group of characters which can be treated as a single unit of information, for example, line numbers.

file

A logically related collection of data treated as a physical entity that occupies one or more blocks on a volume such as a disk or magnetic tape. A file can be referenced by a name assigned by the user. A file normally consists of one or more logical records.

file name

The field preceding an extension in a file specification. This field contains a 1- to 9-character name for a file (1- to 6-characters in CCL).

file specification

A unique name for a file on a mass storage device. It identifies the node, the device, the directory name, the file name, the file type, and the version number under which a file is stored. ("File specification" is a VAX/VMS term.)

fixed line numbers

Line numbers fixed to lines of text in a file. EDT maintains a record of these line numbers during the editing sessions in which you use those files. The line numbers are copied to the file when you end the editing session.

form feed **␣**

Moves the cursor position to the start of a new page. One of the default word delimiters; the default page delimiter.

GOLD key

A key on the keypad. When you press **GOLD** , the alternate function for the next keypad key you press is enabled.

help facility

A set of on-line messages describing the operation of EDT.

increment

To add a quantity to another quantity; the quantity added.

integer

A whole number containing no fractional or decimal part.

journal file

A file containing the data input to the terminal for one editing session.

journaling

The recording of your input during an editing session.

language

A systematic, unambiguous means of communication; a set of representations, conventions, and associated rules used to convey information.

line buffer

A storage area used to store the last line deleted by an EDT delete line operation.

line feed **␣**

Moves the cursor position down one line. One of the default word delimiters. In keypad character editing, deletes the characters from the cursor position to the left word delimiter.

line number

A number used to identify a line of text in a file or in an EDT text buffer.

log in

The process of accessing a computer system; an identification procedure.

macro

A statement that requests EDT to perform a predefined set of instructions.

main text buffer

The default text buffer for keyboard input and for input files, and the source for output files.

memory

A device on which data can be stored and from which it can be retrieved; internal computer storage.

nested parentheses

A parenthetical operation embedded within another parenthetical operation in an expression.

noncontiguous lines

Lines which are not adjacent to one another.

nonprinting character

A character in the computer code set for which there is no corresponding graphic symbol.

null

The character with the ASCII code 000; an absence of information.

null string

A string without content; an empty string represented by adjacent quotation marks.

operating system

Software that controls the execution of computer programs and performs system functions; an integrated collection of programs that supervise computer operation.

parameter

The object of a command. A parameter can be a file specification or a keyword option.

password

The character string assigned to a user to verify the user's access privileges; a code for assuring confidentiality on time-sharing systems.

paste buffer

The default text buffer for EDT cut and paste operations.

program

The complete sequence of instructions, text, and routines necessary for the computer to perform a desired operation.

programming

The process of planning, writing, testing, and correcting the steps required for a computer to solve a problem or perform a desired operation.

prompt

A symbol indicating that the user must provide input.

punctuation

Special language characters such as commas, separators, etc.

qualifier

A command language keyword that modifies the operation of a command. Qualifiers are always preceded by slash (/) characters.

range of lines

The number of lines in a range specification; the range can define single or multiple lines and contiguous or noncontiguous lines.

range specification

A means for defining a string of text.

record

A collection of adjacent items of data treated as a unit. A logical record can be of any length whose significance is determined by the programmer. A physical record is a device-dependent collection of contiguous bytes such as a block on a disk, or a collection of bytes sent to or received from a record-oriented device.

refresh

To update the display with the most current text for the range displayed.

rubout

Synonymous with delete.

screen

The display surface on a display terminal.

screen width

The number of character positions that can be displayed on a line.

search string

A group of characters you define in a command; the object of a search operation.

single line ranges

Ranges which address one line of a text buffer.

string

A set of contiguous items of a similar type; a connected sequence of characters.

string search buffer

A buffer used to store the string being searched for.

substitute buffer

A buffer used to store the string to be substituted for the search string.

symbol

An identifier used to represent a value; a character or group of characters of a language that are used to represent another entity.

syntax

The rules governing command structure in a computer language; the structure of the language.

syntax error

A mistake in your use of the governing command structure.

system

A combination of hardware and software that performs specific processing operations; a collection of components that forms a functional unit.

TAB

Moves the cursor a number of character positions. The default number of positions is eight. One of the default word delimiters.

terminal

The general name for those peripheral devices that have keyboards and video screens or printers. Under program control, a terminal enables you to type commands and text on the keyboard and receive messages on the video screen or printer.

text buffer

An EDT storage area for text (either terminal input or file input).

truncate

To set a limit on the number of characters in a line; characters entered after the limit is reached are used to start a new line.

value

A quantity; the information represented by a data item.

variable

An entity that can assume any of a given set of values; a symbol whose value can change during a program.

version number

The field following the file type in a file specification. It is separated from file type by a period (.) or semicolon (;) and consists of a number that generally identifies it as one version among all files having the identical file specification except for version number. Version numbers are not available on CCL.

Index

Where more than one page number appears for an entry, the main entry is in bold type.

A

Abbreviations of line editing commands, 7-1
ADV (Advance) command, 2-43, **8-5**
ADVANCE function, 2-8, **5-15**
ALL, in range specification, 6-11
AND, in range specification, 6-11
APPEND command, 8-5
APPEND function, 5-38 to 5-40
Arrow keys, 2-10, **5-13**
ASC (ASCII) command, 8-5
ASCII characters, 5-20, 8-5, **B-1**
Asterisk prompt, 2-20, 7-1

B

BACK command, 2-43, **8-6**
BACK SPACE function, 5-19
BACKUP function, 5-15
Baud rates, C-1
BEFORE, in range specification, 6-10
BEGIN, in range specification, 6-8t
BOTTOM function, 2-9, **5-21**
BRIEF qualifier, 7-17
Buffers
 current, 2-35
 MAIN, 1-2, 2-1, **6-5**
 multiple, 1-8, **2-32**
 PASTE, 1-2, 2-1, **6-5**
 rules for naming, 6-6
 SHOW BUFFER command, 9-15
 string search, 5-23
 text, 1-2, **2-1**

C

CCL (Concise Command Language), 2-2
 command line, 4-4
CHANGE command, 2-5, 2-36, 7-1
 for keypad editing, 5-2
 for nokeypad editing, 8-1
Changing case
 in keypad editing, 2-18, **5-48**
 in nokeypad editing, 8-6
 showing current case setting 9-15
CHAR function, 5-16
Character editing, *See CHANGE command*
Characters
 correcting transposition of, 10-6
 flagging uppercase or lowercase, 9-2

CHGC (Change case) command, 8-6
CHNGCASE function, 2-18, **5-48**
 with select range, 2-18
Circumflex (^) and error messages, A-1
Circumflex (^) command, 8-25
CLEAR command, 7-2
Clearing screen with CTRL/W, 5-8
CLI (Command Line Interpreter), 2-2
Command
 ADV (Advance), 8-5
 ADVANCE, 2-43
 APPEND, 8-5
 ASC (ASCII), 8-5
 BACK, 2-43, **8-6**
 CHANGE, 2-5, 2-36, 5-2, 7-1, 8-1
 CHGC (Change case), 8-6
 Circumflex (^), 8-25
 CLEAR, 7-2
 COPY, 2-35, 7-3
 CUT, 8-7
 CUTSEN, 2-43
 D (Delete), 8-9
 DEFINE KEY, 1-9, 7-4, 10-1, **10-2**
 DEFINE MACRO, 7-4
 DEFK (Define key), 8-10, **10-1**
 DELETE, 2-30, 7-6
 EX (Exit), 8-10
 EXIT, 2-28, 2-31, 5-9, 7-8
 EXT (Extended), 8-10
 FILL, 8-11
 FIND, 7-8
 HELP, 1-3, 7-9
 I (Insert), 8-12
 INCLUDE, 2-34, 7-10
 INSERT, 2-21, 2-26, 2-33, 7-11
 INSERT END, 2-27, 2-31
 MOVE, 2-35, 7-11
 NULL (Implied TYPE), in line editing,
 2-23, 7-12
 Null (Implied TYPE), in nokeypad editing,
 8-13
 PASTE, 2-43, **5-33**, 8-15
 PRINT, 7-13
 QUIT, 5-9, 7-14, 8-15
 QUIT/SAVE, 3-2
 R (Replace), 8-15
 REF (Refresh), 8-16
 REPLACE, 2-27, 7-14

Command (Cont.)

RESEQUENCE, 2-31, 2-36, 6-3, 7-15
S/s1/s2 (Substitute), 8-16
SEL (Select), 8-17
SET, 1-9, 9-1t, 9-2
SET CASE, 9-2
SET CURSOR, 9-3
SET ENTITY, 9-4
SET KEYPAD, 2-44, 9-5
SET LINES number, 9-5
SET MODE, 9-5
SET MODE CHANGE, 9-5
SET MODE LINE, 9-5
SET NOKEYPAD, 2-36, 8-1, 9-5
SET NONUMBERS, 9-6
SET NOQUIET, 9-6
SET NOTAB, 9-8
SET NOTRUNCATE, 9-12
SET NOVERIFY, 9-13
SET NOWRAP, 9-14
SET NUMBERS, 9-6
SET QUIET, 9-6
SET SCREEN, 9-6
SET SEARCH, 9-7
SET SEARCH BOUNDED, 9-7
SET SEARCH END, 9-7
SET SEARCH EXACT, 9-7
SET SEARCH UNBOUNDED, 9-7
SET TAB, 9-8
SET TERMINAL, 9-12
SET TRUNCATE, 9-12
SET VERIFY, 9-13
SET WRAP, 5-50, 9-14
SHL (Shift left), 8-18
SHOW, 1-9, 9-1t, 9-15
SHOW BUFFER, 2-34, 9-15
SHOW CASE, 9-15
SHOW CURSOR, 9-16
SHOW ENTITY, 9-16
SHOW KEY, 9-16
SHOW SCREEN, 9-17
SHOW SEARCH, 9-17
SHOW TERMINAL, 9-17
SHOW VERSION, 9-17
SHR (Shift right), 8-18
SN (Substitute next), 8-19
SUBSTITUTE, 2-30, 7-16
SUBSTITUTE NEXT, 7-19
TAB, 8-21
TADJ (Tab adjust), 8-22
TC (Tab compute), 8-23
TD (Tab decrement), 8-24
TI, (Tab increment), 8-24
TOP, 8-24

TYPE, 2-22, 2-26, 2-30, 7-20
UNDC (Undelete character), 8-24
UNDL (Undelete line), 8-25
UNDW (Undelete word), 2-41, 8-24
WRITE, 2-36, 7-22
Command file
 creating, 4-9
 default, 4-9
 EDTINI. EDT, 4-9
 startup, 1-5, 4-1
Command file qualifiers, 1-5, 4-1
 DCL/PDS, 4-2
 MCR/CCL, 4-5
Command function, 2-20, 5-47
 with EXIT, 5-9
 with QUIT, 5-10
Command languages, kinds of, 2-2
Command level
 list of prompts, 2-3t
 returning to, 5-54
Command line, 2-2, 4-1
 DCL/PDS format, 4-2
 DCL/PDS qualifiers, 4-2, 4-3t
 EDT, 2-2
 examples, 4-7
 MCR/CCL format, 4-4
 MCR/CCL qualifiers, 4-5, 4-6t
 and RECOVER qualifier, 3-1
COMMAND qualifier
 DCL/PDS, 4-3
 MCR/CCL, 4-6
Commands
 correcting mistakes, 5-47
 keypad editing, 1-9. *See also Functions*
 line editing, 1-9, 2-20, 7-1
 nokeypad editing, 1-10, 2-36, 8-1, 8-5
 and range specifications, 6-7
 structure of nokeypad, 8-4
 use of, 1-9
Compiled programs, and fixed line numbers, 6-3
Contiguous line ranges, 6-10
Control functions, 5-5
 redefining, 10-1
 using, 1-7
COPY command, 2-35, 7-3
Creating files, 2-20, 4-1
CTRL/A, 5-53
CTRL/C, 5-54
CTRL/D, 5-53
CTRL/E, 5-53
CTRL/K, 5-54, 10-1
 canceling operation of, 10-5
 description, 10-5

CTRL/K (*Cont.*)
 example, 10-6
 CTRL/Q, 2-28
 CTRL/S, 2-28
 CTRL/T, 5-53
 CTRL/U, 5-30, 5-48
 CTRL/W, 5-8
 CTRL/Z, 2-19, 2-22, 5-8, 8-13
 Cursor
 movement in keypad editing, 2-9,
 5-12 to 5-21
 movement in nokeypad editing, 2-37,
 8-13
 setting range of, 9-3
 showing range of, 9-16
 using arrow keys and, 2-12
 Customizing EDT
 with CTRL/K, 5-54
 with DEFINE KEY command, 7-4
 with DEFINE MACRO command, 7-4
 by redefining keys, 10-1
 with SET and SHOW commands, 9-1
 with startup command files, 1-5, 4-9
 CUT command, 8-7
 CUT function, 2-12, 5-33
 example with SELECT and PASTE, 5-35
 reverse, 5-37
 CUTSEN command, 2-43

D

D (Delete) command, 2-40, 8-9
 DCL (Digital Command Language), 2-2, 4-2
 command line, 4-2 to 4-4
 and PDS qualifiers, 4-3t
 Define key (DEFK) command, 8-10
 DEFINE KEY command, 7-4, 10-2
 definition, 1-9, 10-1
 example, 10-4
 DEFINE MACRO command, 7-4
 Defining keys
 description of feature, 1-9
 in keypad editing, 5-54, 10-5
 in line editing, 7-4, 10-2
 in nokeypad editing, 8-10, 10-1
 DEFK (Define key) command, 8-10, 10-1
 DEL C function, 5-26
 DEL EOL function, 2-17, 5-30
 DEL L function, 2-16, 5-32
 DEL W function, 2-15, 5-28
 Delete (D) command, 2-40, 8-9
 DELETE command, 2-30, 7-6
 DELETE function, 2-14, 5-25
 Deleting
 all edits, 5-9, 7-14, 8-15
 characters, 2-14, 5-26, 8-9

lines, 2-15, 5-30, 7-6, 8-9
 words, 2-15, 5-28, 8-9
 Delimiters
 defining delimiters of entities, 9-4
 mixed, 7-17
 string, 2-30, 7-17
 symbols used as, 2-30
 Direction, *See ADVANCE or BACKUP*
 DOWN function, 2-10, 5-13
 DUPLICATE qualifier, 7-3

E

EDT
 command line, 2-2, 4-1
 command line qualifiers, 4-1
 controlling characteristics of, 9-1
 ending a keypad session, 2-19, 5-9
 ending a line editing session, 2-24, 7-8, 7-14
 ending a nokeypad session, 2-44, 8-10, 3-15
 example of command line, 4-1
 features, 1-1
 show version number of, 9-17
 starting a keypad session, 2-5, 5-1
 starting a line editing session, 2-20, 7-1
 starting a nokeypad session, 2-36, 8-1
 starting an editing session, 2-3, 4-1
 EDTINI.EDT, 1-5, 4-9, 4-10
 END, in range specification, 6-8t
 Ending
 a keypad session, 2-19, 5-9
 a line editing session, 2-36, 7-8, 7-14
 a nokeypad session, 2-44, 8-10, 8-15
 Ending an editing session, 5-9
 ENTER function, 5-46
 to correct mistakes, 5-47
 example with COMMAND, 5-47
 Entities
 in keypad editing, 5-16
 in nokeypad editing, 8-2t
 Entity
 BL, 8-2
 BPAGE, 8-3
 BPAR, 8-3
 BR, 8-3
 BSEN, 8-3
 BW, 8-2
 C, 8-2
 EL, 8-2
 EPAGE, 8-3
 EPAR, 8-3
 ER, 8-3
 ESEN, 8-3
 EW, 8-2
 L, 8-2

Entity (Cont.)

- NL, 8-3
- PAGE, 8-3
- PAR, 8-3
- SEN, 8-3
- SR, 8-4
- "string", 8-4
- V, 8-3
- W, 8-2
- [EOB] symbol, 2-4
- EOL function, 5-18
- Error messages
 - list of, A-1
 - purpose of, 3-7
- EXIT command, 2-31, 7-8
 - in keypad editing, 2-20, 5-9
 - in line editing, 2-28, 7-8
 - in nokeypad editing (EX), 2-44, 8-10
- EXIT/SEQUENCE, and VFC format, 6-4
- EXT (Extended) command, 8-10

F

Files

- copying text into, 7-3, 7-22
- multiple, 1-8, 2-32
- restoring, 3-1
- revising, 2-28
- FILL command (nokeypad), 8-11
- FILL function (keypad), 5-49
- FIND command, 7-8
- FIND function (keypad), 5-21
- Finding text
 - in keypad editing, 5-21
 - in line editing, 7-8
 - in nokeypad editing, 8-4

Fixed line numbers, 1-7, 6-3

- assigning, 6-1
- description, 6-1
- to display, 6-4
- example, 6-5

FNDNXT function, 5-23

- repeating, 5-24

FOR, in range specification, 6-10

Form feed, and PAGE, 5-20

FORTTRAN-IV-PLUS language processor, 6-4

Function

- ADVANCE, 2-8, 5-15
- APPEND, 5-38
- BACK SPACE, 5-19
- BACKUP, 5-15
- BOTTOM, 2-9, 5-21
- CHAR, 5-16

- CHNGCASE, 2-18, 5-48
- COMMAND, 2-20, 5-47
- CTRL/A, 5-53
- CTRL/C, 5-54
- CTRL/D, 5-53
- CTRL/E, 5-53
- CTRL/K, 5-54
- CTRL/Q, 2-28
- CTRL/S, 2-28
- CTRL/T, 5-53
- CTRL/U, 5-30
- CTRL/W, 5-8
- CTRL/Z, 5-8
- CUT, 2-12, 5-33, 5-35
- DEL C, 5-26
- DEL EOL, 2-17, 5-30
- DEL L, 2-15, 5-32
- DEL W, 2-15, 5-28
- DELETE, 2-14, 5-25
- DOWN, 2-10, 5-13
- ENTER, 5-46
- EOL, 5-18
- FILL, 5-52
- FIND, 5-21
- FNDNXT, 5-23
- GOLD, 2-5, 5-7
- GOLD integer, 5-7
- GOLD A, 9-8
- GOLD/D, 9-8
- GOLD/T, 9-8
- HELP, 5-7
- LEFT, 2-11, 5-14
- LINE, 5-18
- LINE FEED, 5-28
- OPEN LINE, 5-11
- PAGE, 5-20
- PASTE, 5-33, 5-36
- REPLACE, 5-40
- RESET, 5-8, 5-34
- RIGHT, 2-11, 5-14
- SECTION, 5-20
- SELECT, 2-12, 5-33
- SPECINS, 5-48
- SUBS, 5-44
- TOP, 2-9, 5-21
- UND C, 5-27
- UND L, 2-16, 5-32
- UND W, 5-29
- UP, 2-10, 5-13
- WORD, 2-12, 5-17

Function keys, 5-1

Functions

- and control keys, 5-5
- how to redefine, 10-1

G

GOLD
with alternate functions, 5-5
description, 2-5, 5-7
example, 5-7
instead of control key, 5-53
to repeat functions, 5-7
GOLD integer function, 5-7
GOLD/A function, 9-8
GOLD/D function, 9-8
GOLD/E function, 9-8
GOLD/T function, 9-8

H

Hardcopy terminals
and **SET TERMINAL**, 9-12
and **SHOW TERMINAL**, 9-17
HELP
as line editing command, 1-3, 7-9
function key, 2-5, 5-7
in keypad editing, 1-4, 2-5, 5-7
sample description, 1-3
use of facility, 1-3, 1-4
use of subtopics, 1-4, 7-9
used for keypad key descriptions, 2-5

I

I (Insert) command, 8-12
IAS system prompts and CLI, 2-3t
Implied TYPE (Null)
in line editing, 7-12
in nokeypad editing, 8-13
INCLUDE command, 2-34, 7-10
INSERT command, 2-21, 2-26, 2-33, 7-11
INSERT END command, 2-27, 2-31
Inserting text
INSERT command, 2-26, 2-33, 7-11
INSERT END command, 2-31
in keypad editing, 2-8, 5-2, 5-10
in line editing, 2-26, 7-11
in nokeypad editing (I command), 2-40, 8-12
Integer
to repeat functions, 5-7
used with **GOLD** function, 5-7

J

Journal file
definition, 1-4
editing, 3-4
protecting edits with, 3-1
saving, 3-5, 7-8
successive recoveries, 3-4

JOURNAL qualifier
DCL/PDS, 4-4
MCR/CCL, 4-6
Journaling, 1-4

K

Keypad editing
advantages, 2-4
changing case in, 5-48
commands, 1-9
deleting text in, 5-25
description, 1-7
description of keypad diagrams, 5-6
ending a session, 2-19, 5-9
entering text in, 5-10
essential functions, 5-7
how to use, 2-4, 5-2
reinserting text in, 5-25
setting direction of cursor movement, 5-15
starting, 2-5, 5-1
Keypad functions, 5-3f, 5-4f
ADVANCE, 5-15
alternate, 2-8, 5-5
APPEND, 5-38
BACK SPACE, 5-19
BACKUP, 5-15
BOTTOM, 5-21
CHAR, 5-16
CHNGCASE, 2-18, 2-19, 5-48
COMMAND, 5-9, 5-47
and control keys, 5-53
CTRL/A, 5-53
CTRL/C, 5-54
CTRL/D, 5-53
CTRL/E, 5-53
CTRL/K, 5-54
CTRL/T, 5-53
CTRL/U, 5-30, 5-48
CTRL/W, 5-8
CTRL/Z, 5-8
CUT, 5-33
DEL C, 5-26
DEL EOL, 5-30
DEL L, 5-32
DEL W, 5-28
DELETE, 5-25
DOWN, 5-13
ENTER, 5-46, 5-48
EOL, 5-18
EXIT, 5-9
FILL, 5-49, 5-52
FIND, 5-21
FNDNXT, 5-23
GOLD, 5-5

Keypad functions (*Cont.*)

- GOLD integer, 5-7
- HELP, 5-7
- how to use, 2-5
- LEFT, 5-14
- LINE, 5-18
- LINE FEED, 5-28
- OPEN LINE, 5-11
- PAGE, 5-20
- PASTE, 5-33, 5-36
- QUIT, 5-9
- redefining keys, 10-1
- REPLACE, 5-40
- RESET, 5-8, 5-34
- RIGHT, 5-14
- SECTION, 5-20
- SELECT, 5-33
- setting tabs, 5-53
- SPECINS, 5-48
- standard, 2-8, 5-5
- SUBS, 5-44
- TOP, 5-21
- UND C, 5-27
- UND L, 5-32
- UND W, 5-29
- UP, 5-13
- VT100, 2-6f, 5-4f
- VT52, 2-7f, 5-3f
- WORD, 5-17

Keypad keys

- and functions, 5-1
- redefining functions of, 10-1
- VT100, 2-6f, 5-4f
- VT52, 2-7f, 5-3f

Keypad numbers, 10-3f

L

LAST, in range specification, 6-8t

LEFT function, 2-11, 5-14

Line editing

- abbreviating commands, 7-1
- asterisk prompt, 2-20, 7-1
- creating files in, 2-20
- deleting lines, 7-6
- description, 1-7
- displaying lines, 7-20
- ending a session, 7-8, 7-14
- line numbers, 6-1
- locating text, 7-8
- moving lines, 7-11
- returning to with CTRL/Z, 5-8, 7-2
- returning to with EX, 8-10
- starting, 2-20, 7-2

Line editing commands, 1-9

- CHANGE, 7-1
- CLEAR, 7-2
- COPY, 2-35, 7-3
- DEFINE KEY, 7-4
- DEFINE MACRO, 7-4
- DELETE, 2-30, 7-6
- EXIT, 2-31, 7-8
- FIND, 7-8
- HELP, 1-3, 7-9
- Implied TYPE (Null), 2-23, 7-12
- INCLUDE, 2-34, 7-10
- INSERT, 2-21, 7-11
- MOVE, 2-35, 7-11
- Null, 2-23, 7-12
- PRINT, 7-13
- QUIT, 7-14
- QUIT/SAVE, 3-2, 7-14
- REPLACE, 2-27, 7-14
- RESEQUENCE, 2-31, 2-36, 6-3, 7-15
- SET and SHOW, 9-1
- SUBSTITUTE, 2-30, 7-16
- SUBSTITUTE NEXT, 7-19
- TYPE, 2-22, 7-20
- used in keypad editing, 5-46
- WRITE, 2-36, 7-22

LINE FEED function, 5-28

LINE function, 5-18

Line numbers, 1-7

- assigning new, 7-15
- fixed, 1-7
- in line editing, 6-1
- resequencing, 6-3
- setting display of, 9-6

Lines

- displaying, 9-12
- too long to display on screen, 9-12
- setting display limits, 9-5
- setting length of, 9-14

Locating text

- in keypad editing, 5-21
- in line editing, 7-8
- in nokeypad editing ('string'), 8-4

M

Macro, 7-4, *See also* DEFINE MACRO

Macro-name, defining, 7-4

Macros, definition, 1-5

MAIN text buffer, 1-2

- inserting text into, 2-1, 6-6

MCR (Monitor Console Routine), 2-2

- command line, 4-4

MCR/CCL qualifiers, 4-6t

- Misplaced words, correcting with DEL W and UND W, 5-29
- Mistakes, correcting after COMMAND: prompt, 5-47
- MOVE command, 2-35, 7-11
- Moving the cursor
 - in keypad editing, 2-9, 5-12 to 5-21
 - in nokeypad editing, 2-37, 8-13
- Multiple
 - buffers, 1-8, 2-32
 - files, 1-8, 2-32

N

- NOCOMMAND qualifier
 - DCL/PDS, 4-3
 - MCR/CCL, 4-6
- NOJOURNAL qualifier
 - DCL/PDS, 4-4
 - example, 3-5
 - MCR/CCL, 4-6
- Nokeypad commands
 - ADV (Advance), 8-5
 - APPEND, 8-5
 - ASC (ASCII), 8-5
 - BACK, 8-6
 - CHGC (Change case), 8-6
 - circumflex (^), 8-25
 - CUT, 2-43, 8-7
 - D (Delete), 2-40, 8-9
 - DEFK (Define key), 8-10, 10-1
 - EX (Exit), 8-10
 - EXT (Extended), 8-10
 - FILL, 8-11
 - I (Insert), 2-40, 8-12
 - Null, 8-13
 - PASTE, 2-43, 8-15
 - QUIT, 8-15
 - R (Replace), 8-15
 - REF (Refresh), 8-16
 - S/s1/s2 (Substitute), 2-42, 8-16
 - SEL (Select), 8-17
 - SHL (Shift left), 8-18
 - SHR (Shift right), 8-18
 - SN (Substitute next), 8-19
 - Substitute (S/s1/s2), 8-16
 - TAB, 8-21
 - TADJ (Tab adjust), 8-22
 - TC (Tab compute), 8-23
 - TD (Tab decrement), 8-24
 - TI (Tab increment), 8-24
 - TOP, 8-24
 - UNDC (Undelete character), 8-24
 - UNDL (Undelete line), 8-25
 - UNDW (Undelete word), 2-41, 8-24

- Nokeypad editing
 - buffer definition, 8-4
 - command definition, 8-4
 - command format, 2-37
 - copying text, 8-15
 - count definition, 8-4
 - CTRL/Z and I (Insert), 2-40
 - cursor movement, 2-37, 2-39, 8-2
 - defining entities, 2-39
 - deleting entities, 8-9
 - deleting text, 8-15
 - description, 1-7, 2-36
 - direction definition, 8-4
 - ending a session, 8-10
 - entities, 8-2 to 8-4
 - format, 8-4
 - how to enter, 2-37
 - inserting characters, 2-40
 - inserting text, 8-12
 - moving text into buffers (CUT), 8-7
 - returning to line editing, 2-44
 - reverse search (BACK), 2-42, 8-6
 - search ('string'), 8-4
 - shortening a file, 2-39
 - starting, 2-36, 7-2, 8-1
 - substituting text, 2-42
 - undeleting characters, 8-24
 - undeleting text, 8-25
 - use of commands, 1-10
 - using, 2-37
 - using CTRL/Z with I (Insert), 8-13
- Noncontiguous ranges, 6-11
- NOOUTPUT qualifier (DCL/PDS), 4-3
- Noread Only qualifier
 - NOREAD_ONLY (DCL/PDS), 4-4
 - RO (MCR/CCL), 4-7
- NORECOVER qualifier
 - DCL/PDS, 4-4
 - MCR/CCL, 4-6
- NOTYPE qualifier, 7-17
- Null command (Implied TYPE)
 - in line editing, 2-23, 7-12
 - in nokeypad editing, 8-13
- Numbers
 - displayed in line editing, 6-1
 - fixed line, 1-7, 6-4
 - line, 6-1
 - resequencing of lines, 6-3

O

- OPEN LINE function, 5-11
- Operating system, command line, 2-2
- Original line numbers, 6-8t

ORIGINAL number, in range specification, 6-8t
OUTPUT qualifier (DCL/PDS), 4-3

P

PAGE function, 5-20
PASTE buffer, 1-2, 2-1, 6-6
PASTE command (nokeypad), 2-43, 8-15
PASTE function, 2-12, 5-33
 example with CUT and SELECT, 5-36
PDS (Program Development System), 2-2
 command line, 4-2 to 4-4
Pointer. *See Cursor*
Pointer line and error messages, A-1
PRINT command, 7-13
Programs, and fixed line numbers, 6-3

Q

Qualifiers

BRIEF, 7-17
COMMAND (DCL/PDS), 4-3
COMMAND (MCR/CCL), 4-6
 command file, 1-5, 4-1
DCL/PDS, 4-2
DUPLICATE, 7-3
 examples, 4-7
JOURNAL (DCL/PDS), 4-4
JOURNAL (MCR/CCL), 4-6
 list of DCL/PDS, 4-3t
NOCOMMAND (DCL/PDS), 4-3
NOCOMMAND (MCR/CCL), 4-6
NOJOURNAL, 3-5
NOJOURNAL (DCL/PDS), 4-4
NOJOURNAL (MCR/CCL), 4-6
NOOUTPUT (DCL/PDS), 4-3
NOREAD_ONLY (DCL/PDS), 4-4
NORECOVER (DCL/PDS), 4-4
NORECOVER (MCR/CCL), 4-6
NOTYPE, 7-17
OUTPUT (DCL/PDS), 4-3
 purpose, 4-1
QUERY, 7-3, 7-12, 7-17
 Read Only (MCR/CCL), 4-7
 READ_ONLY (DCL/PDS), 4-4
 RECOVER, 3-1, 3-6
 RECOVER (DCL/PDS), 4-4
 RECOVER (MCR/CCL), 4-6
 RO (MCR/CCL), 4-7
 SAVE, 7-8, 7-14
 SEQUENCE, 6-3, 7-8
 STREAM (CCL), 4-7
 VARIABLE (CCL), 4-7
QUERY qualifier, 7-3, 7-17
 with MOVE, 7-12

QUIT command, 5-9, 7-14, 8-15
 in keypad editing, 2-20, 5-9
 in line editing, 7-14
 in nokeypad editing, 8-15

QUIT/SAVE

 example, 3-4
 in line editing, 7-14
 and RECOVER qualifier, 3-2
 to simulate system failure, 3-2

R

R (Replace) command, 8-15

Range specifications, 2-23

ALL, 6-11
AND, 6-11
BEFORE, 6-10
BEGIN, 6-8t
contiguous line, 6-10
description, 6-7
END, 6-8t
FOR, 6-10
LAST, 6-8t
noncontiguous line, 6-11
number, 6-8t
ORIGINAL number, 6-8t
period (.), 6-8t
REST, 6-10
strings, 6-8t
for text buffers, 6-11
THRU, 2-30, 6-10
WHOLE, 2-22, 6-10

Ranges

contiguous line, 6-10
with MOVE command, 7-11
noncontiguous, 6-11
with SELECT function, 5-33
single line, 6-7, 6-8t
specifiers for noncontiguous, 6-11

Read Only qualifier

 READ_ONLY (DCL/PDS), 4-4
 RO (MCR/CCL), 4-7

RECOVER qualifier, 3-1

 DCL/PDS, 4-4
 example, 3-3 to 3-6
 MCR/CCL, 4-6
 restoring files with, 3-1

Redefining keys

 description of feature, 1-9
 in keypad editing, 5-54, 10-5
 keypad functions, 10-1
 in line editing, 7-4, 10-2
 in nokeypad editing, 8-10, 10-1
 with repeat count, 10-9
 ways of, 10-1

REF (Refresh) command, 8-16
 Refreshing the screen (CTRL/W), 5-8
 Reinserting deleted text
 in keypad editing, 2-12, 2-17, 5-25 to 5-36
 in line editing (MOVE), 7-11
 in nokeypad editing, 2-41, 8-15
 8-24 to 8-25
 Renumbering lines. *See RESEQUENCE command*
 Repeat count, to redefine keys, 10-9
 REPLACE command, 2-27, 7-14
 REPLACE function, 5-40
 RESEQUENCE command, 2-31, 2-36, 6-3, 7-15
 RESET function, 5-8, 5-34
 REST, in range specification, 6-10
 Reverse CUT, 5-37
 Reverse search
 in keypad editing, 5-21
 in nokeypad editing, 2-42
 Reverse SELECT, 5-52
 Reverse video, 2-12, 5-33
 Revising files with buffers, 2-1
 RIGHT function, 2-11, 5-14
 RO qualifier (MCR/CCL), 4-7
 RSTS/E system prompts and CLI, 2-3t
 RSX/11M-PLUS system prompts and CLI, 2-3t
 RSX-11M system prompts and CLI, 2-3t

S

S/s1/s2 (Substitute) command
 description, 8-16
 example, 2-42
 SAVE qualifier, 7-14
 in line editing, 7-8
 with QUIT, 3-2, 3-4
 Saving edits, 5-9, *See also EXIT command*
 Screen
 refreshing, 5-8
 shifting image of, 8-18
 showing parameters of, 9-17
 width (SET SCREEN), 9-6
 Scrolling, 2-28
 Search
 setting direction of, 2-42, 5-21, 8-5, 8-6
 showing parameters of, 9-17
 SECTION function, 5-20
 SEL (Select) command, 8-17
 Select (SEL) command, 8-17
 SELECT function, 2-12, 5-33
 example with CUT and PASTE, 5-34
 reverse, 5-52
 SELECT range, marking text with, 2-12, 5-34

SEQUENCE qualifier
 description, 6-3
 and fixed line numbers, 6-4
 in line editing, 7-8
 Sequencing lines, *See RESEQUENCE command*
 SET and SHOW parameters, 9-1t
 SET commands
 description, 1-9, 9-2
 function of, 9-1
 list of, 9-1t to 9-2t
 SET CASE, 9-2
 SET CURSOR, 9-3
 SET ENTITY, 9-4
 SET KEYPAD 2-44, 9-5
 SET LINES number, 9-5
 SET MODE, 9-5
 SET MODE CHANGE, 9-5
 SET MODE LINE, 9-5
 SET NOKEYPAD, 2-36, 7-2, 8-1, 9-5
 SET NONUMBERS, 9-6
 SET NOQUIET, 9-6
 SET NOTAB, 9-8
 SET NOTRUNCATE, 9-12
 SET NOVERIFY, 9-13
 SET NOWRAP, 9-14
 SET NUMBERS, 9-6
 SET QUIET, 9-6
 SET SCREEN width, 9-6
 SET SEARCH, 9-7
 SET SEARCH BOUNDED, 9-7
 SET SEARCH END, 9-7
 SET SEARCH EXACT, 9-7
 SET SEARCH UNBOUNDED, 9-7
 SET TAB, 9-8
 SET TERMINAL, 9-12
 SET TRUNCATE, 9-12
 SET VERIFY, 9-13
 SET WRAP, 5-50, 9-14
 SET parameters, and SHOW parameters 9-1t
 Setting control characteristics, with SET and SHOW, 9-1
 Setting tabs
 in keypad editing, 5-53
 in nokeypad editing, 8-21
 SHL (Shift left) command, 8-18
 SHOW commands
 description, 1-9, 9-15
 function of, 9-1
 list of, 9-1t to 9-2t
 SHOW BUFFER, 2-34, 6-5, 7-2, 9-15
 SHOW CASE, 9-15
 SHOW CURSOR, 9-16
 SHOW ENTITY, 9-16

SHOW commands (*Cont.*)

- SHOW KEY, 9-16
- SHOW SCREEN, 9-17
- SHOW SEARCH, 9-17
- SHOW TERMINAL, 9-17
- SHOW VERSION, 9-17
- Show parameters, and SET parameters, 9-1t
- SHR (Shift right) command, 8-18
- Simulating a system failure, 3-2
- Single line ranges, 6-7, 6-8t
- Slashes
 - with DCL/PDS qualifiers, 4-3
 - with MCR/CCL qualifiers, 4-5
- SN (Substitute next) command, 8-19
- Special characters
 - as delimiters, 2-30
 - in nokeypad substitutions, 8-16
- SPECINS function, 5-48
- Starting
 - an EDT editing session, 2-1, 4-1
 - keypad editing, 2-5, 5-1
 - line editing, 2-20, 6-1, 7-1
 - nokeypad editing, 2-36, 8-1
- Startup command files
 - creating, 4-9
 - definition, 1-5
 - EDTINI.EDT, 1-5, 4-9
 - to specify mode, 9-5
 - use of, 4-1
- STREAM qualifier (CCL), 4-7
- String search buffer, example, 5-23
- Strings
 - delimiters, 7-17
 - replacing, 7-14
 - search (keypad), 5-21 to 5-25
 - search (line editing), 7-8
 - search (nokeypad 'string'), 8-4
- Strings, substituting
 - in keypad editing, 5-46
 - in line editing, 2-30, 7-16 to 7-19
 - in nokeypad editing, 2-42, 8-16
- SUBS function, 5-44
- SUBSTITUTE command, 2-30, 7-16
- SUBSTITUTE NEXT command, 7-19
- Substituting text
 - in keypad editing, 5-44, 5-46
 - in line editing, 2-30, 7-16, 7-19
 - in nokeypad editing, 2-42, 8-16
 - with redefined key, 10-8
- Subtopics, HELP, 1-4, 7-9
- System failure
 - restoring edits after, 3-2
 - simulating, 3-2

T

Tabs

- adjusting level of, 5-53, 8-23
- and CONTROL keys, 5-53
- in keypad editing, 5-53
- setting parameters of, 9-8
- setting position of, 8-21, 8-24
- tab adjust (TADJ) command, 8-22
- TAB command, 8-21
- tab compute (TC) command, 8-23
- Tab decrement (TD) command, 8-24
- Tab increment (TI), 8-24
- TADJ (Tab adjust) command, 8-22
- TC (Tab compute) command, 8-23
- TD (Tab decrement) command, 8-24

Terminals

- hardcopy, 9-12
- using SHOW TERMINAL command, 9-17

Terminals, video. *See VT52 or VT100*

Text buffers, 1-2

- automatically created, 6-6
- copying files from, 7-13, 7-22
- copying files into, 7-10
- creating additional, 6-6
- deleting contents of, 7-2
- for inserting text, 2-32, 7-11
- listing, 9-15
- MAIN, 1-2, 2-1, 6-5
- moving cursor throughout, 5-20
- for moving entities, 8-7
- PASTE, 1-2, 2-1, 6-5
- and range specifications, 6-11
- restrictions, 6-6
- rules for naming, 6-6
- use of, 2-1

Text entities, 8-2t

- THRU, in range specification, 2-30, 6-10
- TI (Tab increment) command, 8-24
- TOP command, 8-24
- TOP function, 2-9, 5-21
- TYPE, Implied (Null)
 - in line editing, 7-12
 - in nokeypad editing, 8-13
- TYPE command, 2-22, 2-26, 2-30, 7-20

Typographical errors, correcting with
DEL C and UND C, 5-27

U

- UND C function, 5-27
- UND L function, 2-16, 2-17, 5-32
- UND W function, 5-29
- UNDC (Undelete character) command, 8-24

- Undeleting
 - buffer contents after system failure, 3-1
 - characters (keypad), 5-27
 - characters (nokeypad), 8-24
 - lines (keypad), 2-16, 2-17, 5-32
 - lines (nokeypad), 8-25
 - with PASTE (keypad), 5-36
 - with PASTE (nokeypad), 8-15
 - with REPLACE (line editing), 7-14
 - words (keypad), 5-29
 - words (nokeypad), 8-24
- UNDL (Undelete line) command, 8-25
- UNDW (Undelete word) command, 2-41, 8-24
- UP function, 2-10, 5-13

V

- VARIABLE qualifier (CCL), 4-7
- VAX/VMS, 2-2. *See also* DCL
- VAX/VMS system prompts and CLI, 2-3t
- Version numbers
 - and SHOW VERSION commands, 9-17
 - systems without, 4-5
- VFC records, 6-3
- Video terminals. *See* VT52 or VT100
- VT100
 - and CHAR function, 5-16
 - and EDT editing, 1-9
 - and FILL function, 5-49
 - and HELP text, 1-4f

- keyboard and keypad, 1-6f
 - and keypad editing, 1-2, 2-4, 5-4f
- keypad numbers, 10-3f
 - and nokeypad editing, 2-36, 8-1
 - and select ranges, 2-12, 5-33
 - and SET KEYPAD, 9-5
 - and SET NOKEYPAD, 9-5
 - and SET SCREEN, 9-6
 - and SET TERMINAL, 9-12
 - and SHOW TERMINAL, 9-17
- VT52

- and EDT editing, 1-9
- and FILL function, 5-49
- and HELP text, 1-4
- and keypad editing, 1-2, 2-4, 5-3f
- keypad numbers, 10-3f
 - and nokeypad editing, 2-36, 8-1
 - and select ranges, 2-12, 5-33
 - and SET KEYPAD, 9-5
 - and SET NOKEYPAD, 9-5
 - and SET SCREEN, 9-6
 - and SET TERMINAL, 9-12
 - and SHOW TERMINAL, 9-17

W

- WHOLE, in range specification, 6-10
- Width of text, limiting, 5-49
- WORD function, 2-12, 5-17
- Word wrap, setting, 5-50
- WRITE command, 2-36, 7-22

Reader's Comments

Note: This form is for document comments only. Digital will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement. _____

Did you find errors in this manual? If so, specify the error and the page number. _____

Please indicate the type of user/reader that you most nearly represent.

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Other (please specify) _____

Name _____ Date _____

Organization _____

Street _____

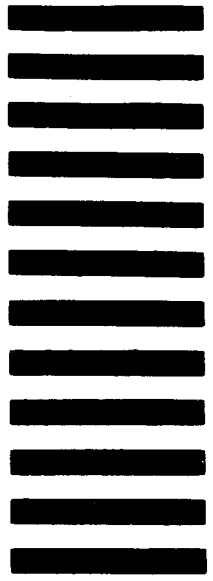
City _____ State _____ Zip Code _____
or
Country _____

-- -- --Do Not Tear - Fold Here and Tape -- -- --

digital



No Postage
Necessary
if Mailed in the
United States



BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO.33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

ATTN: Commercial Engineering Publications MK1-2/ H3
DIGITAL EQUIPMENT CORPORATION
CONTINENTAL BOULEVARD
MERRIMACK N.H. 03054

-- -- -- Do Not Tear - Fold Here and Tape -- -- --

Cut Along Dotted Line