

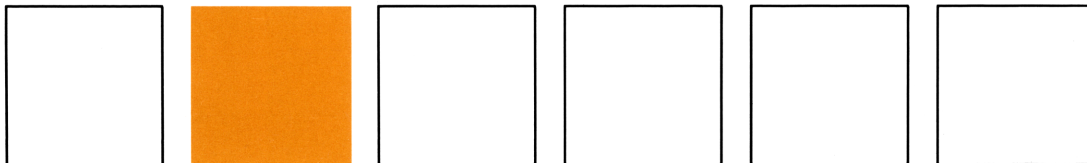


HP 82485A

Text Editor

Owner's Manual

For the HP-71



Notice

Hewlett-Packard Company makes no express or implied warranty with regard to the keystroke procedures and program material offered or their merchantability or their fitness for any particular purpose. The keystroke procedures and program material are made available solely on an “as is” basis, and the entire risk as to their quality and performance is with the user. Should the keystroke procedures or program material prove defective, the user (and not Hewlett-Packard Company nor any other party) shall bear the entire cost of all necessary correction and all incidental or consequential damages. Hewlett-Packard Company shall not be liable for any incidental or consequential damages in connection with or arising out of the furnishing, use, or performance of the keystroke procedures or program material.



Text Editor

Owner's Manual

For Use With the HP-71

March 1984

82485-90001

Introducing the Text Editor

Your HP 82485A Text Editor can be used to create, view, change, and print source program listings, text files, memos, form letters, personal letters, short reports, and other documents. In fact, the Text Editor enables you to manipulate any text file in your HP-71 memory.

Using the Text Editor, you can:

- Create new text files.
- Update text files by editing, deleting, and inserting lines of text.
- Search for and edit text.
- Copy or move sections of one file to another file.
- Print or list text files.

As you compose a text file, you can include (*embed*) text formatting commands in it. When you execute the format command, these embedded commands determine how your formatted, printed document will look.

Refer to “How To Use This Manual,” page 7, for information on how to get the most out of this manual.

Contents

How To Use This Manual	7
Section 1: Getting Started	9
Installing and Removing the Text Editor Module	9
Using the Keyboard Overlay	10
How to Start the Text Editor	11
Creating a File	11
Entering Text	12
Moving Around in a File	14
Inserting Lines Into a File	15
Exiting From a File	16
Section 2: Text Editor Commands	17
Parameters for the Text Editor Commands	17
The T (Text) Command	19
The E (Exit) Command	22
The L (List) and P (Print) Commands	22
The C (Copy) and M (Move) Commands	24
The I (Insert) Command	25
The D (Delete) Command	27
The S (Search) and R (Replace) Commands	28
Special Patterns	33
The F (Format) Command	35
Concatenating Commands	36
The H (Help) Command	36
Section 3: Text Formatting Commands	39
General Syntax for Commands	39
The Command Character (^)	39
Parameters	40
Text Input and Output	41
Typing Conventions in This Section	41

Text Formatting Modes	41
Filling Lines and Justifying Lines (^FI, ^JU)	41
Copying Lines and Centering Lines (^CO, ^CE)	44
Testing for the End of the Page	46
Indenting Lines	46
Starting a New Paragraph (^PA)	46
Indenting by Tab to a Column (^TA)	47
Formatting a Page	48
Setting Margins (^MA)	49
Page Length (^PL)	50
Line Spacing (^SP)	52
Skipping Lines (^SK)	53
Page Numbering (^PN)	53
Advancing the Page (^AD)	56
Using the Escape Command (^ES)	56
Formatting Combinations of Text Files	56
Merging Files (^ME)	57
Using a Distribution List (^DL)	58
Appendix A: Owner's Information	63
Limited One-Year Warranty	63
Service	64
When You Need Help	67
Appendix B: Error and Status Messages	69
Text Editor Messages	69
Text Formatting Messages	70
Displayed Messages	71
Printed Messages	71
Appendix C: Text Editor Module BASIC Language Enhancements	73
Appendix D: Text Editor Filenames	77
Appendix E: Using Printer Escape Sequences	79
Appendix F: Text Formatting Subprograms	81
Command Summary and Index	83
Subject Index	91

How To Use This Manual

This manual assumes that you are familiar with sections 1, 6, 7, and 13 of your *HP-71 Owner's Manual*. Specifically, you will need to know the basics for operating your HP-71. If you want to use a printer, you need to be familiar with the *HP 82401A HP-IL Interface Owner's Manual* so that you know how to connect, assign, and operate your printer (sections 1 and 2).

This manual is arranged so that you begin using your Text Editor in section 1, "Getting Started." Section 1 outlines the commands to create a file, move around in a file, and close a file. Read and work through the examples in this section to learn the basic operation of the Text Editor. The command syntax learned in section 1 is the same syntax used in later sections.

The Text Editor commands are described in section 2, "Text Editor Commands." Examples are given to show how to use the commands.

Section 3, "Text Formatting Commands," describes how to embed text formatting commands in your text. These embedded commands determine how your printed copy will appear. Formatting commands do such things as start new paragraphs, set margins, number pages, and generally provide the tools you need to make neat, well-structured documents. Examples are given to illustrate the use of each command.

Later, when you need brief descriptions and reminders for your Text Editor commands, refer to the *HP 82485A Quick Reference Guide* or the "Command Summary and Index" at the back of this manual.

There are several appendices for your reference:

- Appendix A, "Owner's Information," includes warranty and service information.
- Appendix B is "Error and Status Messages." If the Text Editor can't execute a certain command, an error message is displayed. Refer to appendix B for an explanation of the messages that the Text Editor can display. (See the *HP-71 Owner's Manual* for explanation of other messages.)
- Appendix C, "Text Editor Module BASIC Language Enhancements," explains the nine Text Editor keywords that can be used in BASIC language programs, or from the keyboard.
- Appendix D, "Text Editor Filenames," lists the filenames used in the Text Editor.
- Appendix E, "Using Printer Escape Sequences," explains how to put printer escape sequences into your file.
- Appendix F, "Text Formatting Subprograms," describes the text formatting subprogram and four read and print subprograms that you can use as "hooks" into the text formatting subprogram.

Section 1

Getting Started

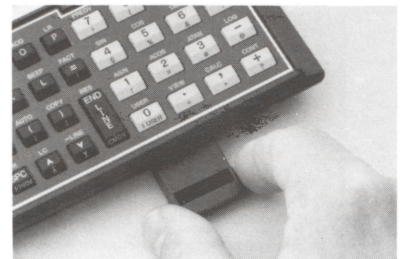
Installing and Removing the Text Editor Module

The Text Editor module can be plugged into any one of the four ROM ports on the front edge of the computer.

CAUTIONS

- Be sure to turn off the HP-71 (press **[f]** **[OFF]**) before installing or removing the module.
- If you have removed a module to make a port available for the Text Editor module, before installing the Text Editor module turn the computer on and then off to reset internal pointers.
- Do not place fingers, tools, or other foreign objects into any of the ports. Such actions could result in minor electrical shock hazard and interference with pacemaker devices worn by some persons. Damage to port contacts and internal circuitry could also result.

To insert the module, orient it so that the label is right side up, hold the computer with the keyboard facing up, and push the module into the slot until it snaps into place. During this operation be sure to observe the precautions described above.



To remove the module, use your fingernail to grasp the lip on the bottom of the front edge of the module and pull it straight out of the port. Install a blank module in the port to protect the contacts inside.

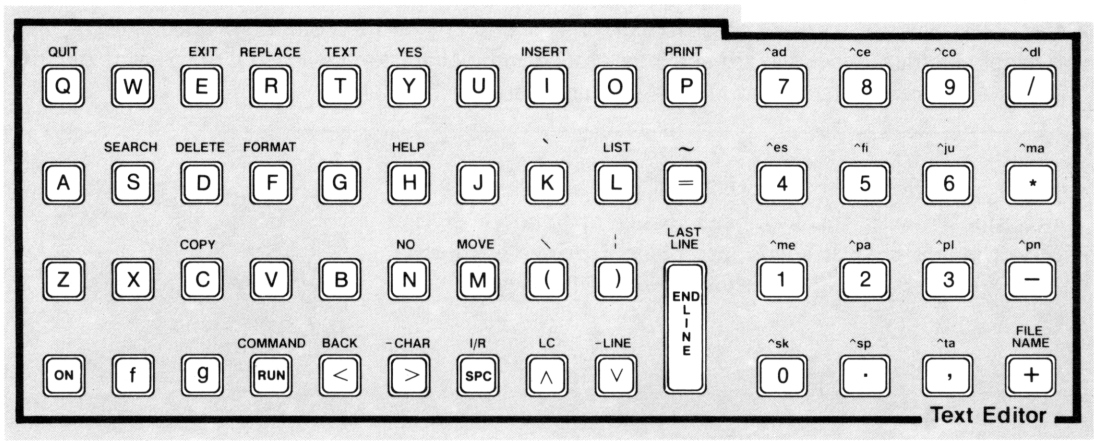
Using the Keyboard Overlay

When you run the Text Editor, many of the keys are used by the program. The keyboard overlay shows the positions of the Text Editor commands and the text formatting commands on the keyboard.

The Text Editor commands are listed as a reminder of the command letters and are printed in white (indicating the primary function of the key). These commands are active only when the Command Prompt is in the display. The text formatting commands are printed in yellow, indicating that you must press **f** before you press the operation key. These are primarily typing aids, putting the three character command in the display so you don't have to type it.

If you have redefined the keyboard, your redefinitions are copied by the Text Editor and stored in the file `EDUKEYS`. The Text Editor keyboard is then superimposed on your own keyboard. Unless one of the Text Editor keys happens to be one that you have defined, your defined keys are still available to you. When you exit the Text Editor, the Text Editor keys are purged, and your defined keys are re-stored.

Below is a facsimile of the HP-71 keyboard with the overlay in place.



How to Start the Text Editor

Now that the Text Editor module is in place, put the keyboard overlay on the keyboard and turn on your HP-71.

Some basic Text Editor operations are introduced in this section to help you understand how to use the Text Editor. This section is organized as follows:

1. A task is defined and you are told how to do it.
2. You practice doing it yourself.
3. An example of the display is shown so that you can check your results.

In this section the following basic operations are described:

1. Creating a file
2. Entering text
3. Moving around in a file
4. Inserting text in a file
5. Exiting the Text Editor

We recommend that you follow through and do the examples in this section so that you become familiar with the general operation of the Text Editor.

Creating a File

The Text Editor is used to create a new file or modify an existing file. You call the Text Editor by entering `EDTEXT filename` `END LINE`. If you call `EDTEXT` with an existing filename, that file will be opened for editing. If you use a new name, the new file will be created. If no filename is specified, you will get the error `Invalid Filespec`. Refer to section 6 of your *HP-71 Owner's Manual* and appendix D of this manual for information about valid filenames and reserved words.

To start working with the Text Editor, create a file named `SAMPLE` using the `EDTEXT` keyword.

Input/Result

```
EDTEXT SAMPLe END LINE
```

Since there is no existing file called `SAMPLE`, `EDTEXT` will create an empty file with that name.

```
Eof, Cmd: █
```

The display shows that the *current line* of the file is “end of file” (`Eof`) and that the Text Editor is ready for a command (indicated by `Cmd:`).

The `Cmd:` display is the Command Prompt. Displayed with the Command Prompt is the current line number in the file. Since the current line is `Eof`, there is no line number in the display.

Lines of text are numbered beginning at 1. The numbers appear in the display to give you reference points in a file, but these numbers are not stored in the file. The importance of line numbers will become clear when you begin to use the different capabilities of the Text Editor.

When you are in the Text Editor, the filename is always accessible prior to line 1, although it is not stored in the file. Because `SAMPLE` is still empty, the `Eof` line directly follows the filename. If you press `[F]` now, the Text Editor will display the edit filename. You can display the edit filename from any place in the file by pressing `[F][+]` when the Command Prompt is in the display.

Input/Result

`[F][+]`

Press `[F]`, then press and hold `[+]`.

```
Filename: SAMPLE
```

The Text Editor displays the name of your edit file while you hold down the `[+]` key.

Release the key.

```
Eof, Cmd: █
```

When you release `[+]`, the Command Prompt is displayed again.

Entering Text

Now enter some lines of text. Use the command `T` (the `T` can be upper or lower case) to begin entering text. (Each of the commands you use here are explained in more detail later.)

Input/Result

`T [END LINE]`

```
█
```

The Text Editor gives you a blank line to enter your text.

The Text Editor sets the HP-71 to lower case mode because this is the most common mode for text applications. You can, of course, enter upper case or lower case letters. Use `[F][LC]` and `[G]` to move back and forth between the two modes.

If you make an error before you press `[END LINE]`, use `[←]` to back up and correct the error. The current line is not entered into the file until you press `[END LINE]`. If you have already pressed `[END LINE]`, then discover you made an error, don't worry about it now; you will learn how to correct the contents of a file in section 2.

Now enter the following line of text:

Input/Result

Jack and Jill went up the hill

Jill went up the hill■

Before you press **END LINE**, your display should look like the display at left.

END LINE

■

After you press **END LINE**, the Text Editor gives you another blank line.

Now enter another line of text, then press **END LINE** to store the text and advance to another blank line:

to fetch a pail of water. **END LINE**

For this demonstration, let us “forget” to put in a line of the rhyme and next enter:

and Jill came tumbling after. **END LINE**

Now, return to the Command Prompt by pressing **RUN** or **ATTN**.

Input/Result

RUN

Eof, Cmd:■

The Text Editor prompts for a command.

Notice that the current line is still Eof. Each time you pressed **END LINE**, the line you entered was *added* to the file ahead of Eof. Each new line had a line number that was advanced by one—starting at line 1.

Moving Around in a File

You can move to any line in a file by entering a line number and pressing **END LINE**. For example, if you want to move to line 2, enter 2 **END LINE**.

Input/Result

2 **END LINE**

Line 2, Cmd:■

The current line is now line 2 of your file and the Text Editor prompts for a command.

If you press **END LINE**, line 2 will scroll through the display. Note that you can control the scrolling speed by changing the delay rate. Refer to section 1 of your *HP-71 Owner's Manual*.

Input/Result

END LINE

Hold for a few seconds.

← fetch a pail of water.

Line 2 scrolls through the display to the end of the line. The left arrow annunciator indicates that the line of text extends to the left.

When you release **END LINE**, the Command Prompt reappears.

Line 2, Cmd:■

In addition to entering a line number, you can use the **↑** and **↓** keys to move to other lines of the file. To move to a line with a smaller line number, press **↑**. To move to a line with a larger line number, press **↓**. You can also move to the beginning or the end of a file by pressing **g↑** or **g↓**.

Input/Result

↓

1 came tumbling after.

Pressing **↓** displays the next larger line number. The line of text scrolls through the display. The end of the line will remain displayed as long as **↓** is held down.

Line 3, Cmd:■

The Command Prompt appears when you release **↓**.

G **↑**

```
Jill went up the hill
```

Press **G** **↑** to move to the first line of the file. The line of text will scroll through the display until the last 22 characters are shown in the display.

```
Line 1, Cmd:█
```

The Command Prompt reappears when you release **↑**.

To display the name of the edit file, press **↑** when you are at line 1.

Input/Result

↑

```
Filename: SAMPLE
```

Hold the key down for a few seconds and the filename will be displayed.

```
Line 1, Cmd:█
```

When **↑** is released, line 1 scrolls through the display, then the Command Prompt appears again.

Inserting Lines Into a File

Do you remember that line we “forgot” to enter in our `SAMPLE` file? Now we will add this line to the file. To insert the line, we will use the `I` (for insert) command. When you insert a line, it is always inserted *before* the current line in the file.

Input/Result

3I **END LINE**

3I tells the Text Editor to begin inserting new lines in front of line 3.

```
and Jill came tumbling
```

Line 3 appears in the display until you begin entering the new line.

Now enter the new line of text:

Input/Result

```
Jack fell down and broke his
crown,
```

```
and broke his crown, █
```

END LINE

Again, the end of the line is displayed until you press **END LINE**.

Pressing **END LINE** inserts the new line into the file.

```
and Jill came tumbling
```

Old line 3 appears in the display after you release **END LINE**.

Return to the Command Prompt by pressing **RUN** or **ATTN**.

Input/Result

RUN

```
Line 4, Cmd: █
```

The current line is now line 4. Line 3 became line 4 when you inserted the new line.

You can now check the text in line 4 by pressing **END LINE**. If you press **↑**, you will see the new text in line 3. You can use your cursor keys to display all four lines of text. In section 2, you will learn how to display all or part of your text in a sequence.

To complete an editing session, you must close your file by using the E (for exit) command (described next).

Exiting From a File

The E command ends the editing session.

To end this editing session:

Input/Result

E **END LINE**

```
Done: SAMPLE
```

The **SAMPLE** file is closed and the editing sessions ends.

Text Editor Commands

This section describes the following Text Editor operations:

- Parameters for the Text Editor commands.
- Entering text in a file using the `T` (text) command.
- Exiting from a file using the `E` (exit) command.
- Reviewing the contents of a file using `L` (list) and `P` (print).
- Copying or moving a line or part of a file using `C` (copy) or `M` (move).
- Adding lines to a file using `I` (insert).
- Deleting lines from a file or moving text from one file to another using `D` (delete).
- Searching for a string expression within a file using `S` (search).
- Searching for a string, then replacing the string, using `R` (replace).
- Using special patterns to enhance `S` (search) and `R` (replace).
- Formatting and printing a file using `F` (format).
- Entering more than one command on a line.
- Reviewing the syntax for the Text Editor commands using `H` (help).

Parameters for the Text Editor Commands

Some Text Editor commands require additional information, called *parameters*—such as line numbers or a filename—to perform their functions. When an optional parameter value is not specified, the default value is used.

These parameters are identified in syntax diagrams for each command. For example, the syntax diagram below is for the L (list) command:

[*beginning line number* [*ending line number*]] L [*number of lines*] [*N*]

In the syntax diagram above:

- The L is the list command specifier. All command specifiers may be entered in upper case or lower case, as can filenames.
- *Beginning line number* and *ending line number* are optional parameters, because they are enclosed in square brackets. They specify the first and last line of the lines to be listed. Note that you can't specify the *ending line number* without first specifying the *beginning line number*. The default values for these parameters are listed with the syntax diagram for each command, and also in the "Command Summary and Index."
- The *number of lines* is an optional parameter that allows you to specify the number of lines to be listed.
- The N is another optional parameter that causes the line number of each line to be displayed with the line.

Syntax diagrams like this one will be used throughout this manual to illustrate the general form of each Text Editor command.

In these syntax diagrams:

- Items enclosed in square brackets ([]) are *optional parameters*. Note that some optional parameters are nested within others. This indicates that the parameter in the outer pair of brackets must be present *before* the parameter in the inner pair can be included.
- Items shown in DOT MATRIX text (for example, L and N, above) must appear exactly as shown (either upper or lower case is acceptable).
- There are two substitute characters that can be used for any line number parameter:
 - The period indicates the current line.
 - The pound sign indicates the last line in the file.
- Two adjacent numeric parameters must be separated by a space. A numeric parameter next to an alphabetic parameter (or vice versa) needs no separating space. A comma can be used in place of the space in Text Editor commands, but not in text formatting commands.

The T (Text) Command

```
[line number] T
```

Default value: *line number* = current line

The T (text) command is the “workhorse” of the Text Editor. When you enter text mode, the current line appears in the display with the cursor at the beginning of the line. All of the HP-71 editing keys are active in text mode. For example, you can use the arrow keys to move to each line of text and examine the characters. No changes will be made unless you press **END LINE** after you overwrite a character, a word, or the entire line.

Changes made to a line are stored in the file when you press **END LINE**. If you make changes, but move to another line using one of the arrow keys (**↓** or **↑**) *before* you press **END LINE**, your changes will *not* be entered. When you press **END LINE** to store your changes, the *current line* will be advanced. If you are at the end of the file, a blank line will appear in the display. You can enter text on the blank line and by pressing **END LINE**, *append* it to the file. Thus, new lines can be added to an existing file.

You can recall the previous line to the display (either a command or a line in the file) by pressing **9** **END LINE**. To view the current line in the file when you have a Command Prompt, press **END LINE**.

The Text Editor will not go into text mode on a line longer than 96 characters. The error message **Line Too Long** will be displayed, followed by the Command Prompt.

Example: Edit the nursery rhyme file you created in section 1 by entering **EDTEXT SAMPLE**.

Input/Result

```
EDTEXT SAMPLE END LINE
```

```
Line 1, Cmd:■
```

Enter the command **3T** to move to line 3 of your text.

Input/Result

```
3T END LINE
```

```
Jack fell down and bro
```

Change the name Jack to Jill by overwriting Jack with Jill. Move the cursor to his in the display and overwrite his with her. Now, move the cursor to crown and overwrite the first three letters with the word arm.

Input/Result

```
own and broke her armw
```

Making corrections is not always as easy as making the direct substitutions shown in the previous example. Corrections often have to be made that require you to delete or add characters. Characters are removed using **f** **-CHAR** and added using **f** **I/R** to insert characters where there is no space.

The cursor is now at the first of two extra letters after arm. Delete these extra letters by holding down the **f** key and pressing **-CHAR** two times.

Input/Result

f **-CHAR** **-CHAR**

```
own and broke her arm,
```

The extra letters are deleted and the spaces are closed.

END LINE

Store the correction.

```
and Jill came tumbling
```

The next line is displayed.

Move the cursor to Jill in the fourth line of the rhyme, delete Jill, then insert Jack.

Input/Result

f **-CHAR** **-CHAR** **-CHAR** **-CHAR**

```
and  came tumbling aft
```

The word Jill is deleted and the text shifts to the left to close the space.

f **I/R**

Changing from the *replace* cursor (a box) to the *insert* cursor (an arrow) enables you to make the insertion.

```
and  came tumbling aft
```

The *insert cursor* appears. Now insert the name Jack.

Jack

```
and Jack came tumbling
```

END LINE

```
■
```

The display opens to make room for the new characters as you type them.

Finally, store the correction. The insert cursor goes away when you press **END LINE** or **f I/R**.

A blank line appears for the addition of more text.

The changes you made in your text were stored each time you pressed **END LINE**. If you had pressed either of the vertical arrow keys, the change would not have been saved. To illustrate this, enter a few meaningless characters in the blank line and press **↑**.

Input/Result

zzz **↑**

Enter a few meaningless characters, then go to the previous line by pressing **↑**.

```
and Jack came tumbling
```

The previous line appears in the display.

Note that the change from Jill to Jack was stored in this line when you pressed **END LINE**. The zzz addition was not saved, however. You can recall the zzz line by pressing **f END LINE**.

Input/Result

f **END LINE**

```
zzz ■ Jack came tumbling
```

The zzz line appears in the display and writes over the current line display. The cursor appears at the *end* of the zzz line you recalled.

↓

Press **↓** to see the actual contents in line 5 (zzz would have been in line 5 had you pressed **END LINE** after typing in zzz).

```
■
```

After a brief Eof display, a blank line appears, because there is no line 5.

You can also verify that this is the end of the file by pressing **[RUN]** or **[ATTN]**.

Input/Result

[RUN]

```
Eof, Cmd:■
```

The Text Editor returns to the Command Prompt.

The E (Exit) Command

```
E
```

This command ends the editing session. The text remains in memory under the filename used when you entered the Text Editor.

If you decide not to keep this file, purge it following the instructions in your *HP-71 Owner's Manual*, section 6.

All features of your HP-71 return to the original user environment when you exit, except for the display window. Regardless of how the window was set before you ran the Text Editor, it will be set to **WINDOW 1** on exiting. Refer to section 7, “Protected Display Fields,” in your *HP-71 Owner's Manual* for more information about the **WINDOW** statement.

If you exit the Text Editor through some means other than the **E** command, all the Text Editor key assignments will be active and your file will be open. To get back into the Text Editor so you can exit correctly, type **CONT [END LINE] [RUN]**, or get out of User mode by pressing **[9][USER]** then press **[f][CONT] [RUN]**.

The L (List) and P (Print) Commands

The **L** (list) and **P** (print) commands are similar to one another. **L** will cause the specified lines of text to be displayed consecutively on the current display device (usually the display window or a monitor). If you have an HP 82401A HP-IL Interface installed and a printer assigned, **P** will cause the specified lines to be printed. Without a printer present, **P** will respond like **L**. You must have the Command Prompt in your display before you enter a Text Editor command, like **L** or **P**.

[*beginning line number* [*ending line number*]] L [*number of lines*][N]

[*beginning line number* [*ending line number*]] P [*number of lines*][N]

Default values: *beginning line number* = current line
ending line number = last line

After listing or printing, the *current line* will be the line after the *ending line number*. The following examples show some L and P commands with parameters:

L	List from the current line to the end of the file.
.L10	List, starting with the current line, 10 lines or to the end of the file, whichever comes first.
3 9 L N	List from line 3 to line 9 with line numbers.
1 L20N	List, with line numbers, the first 20 lines or the entire file, whichever comes first.
P	Print from the current line to the end of the file.
.P5N	Print, with line numbers, five lines or to the end of the file, whichever comes first, starting at the current line.
1P N	Print the entire file with line numbers.

To list your file SAMPLE, you could use the line numbers 1 and 4 as the first two parameters, or you could use # as the second parameter, or you could leave out the ending line number parameter because the default value is the last line of the file. Now, list your file with line numbers.

Input/Result

EDTEXT SAMPLE ENDLINE

1#LN ENDLINE

Call for the SAMPLE file.

List the SAMPLE file from line 1 to Eof with line numbers.

```
1:Jack and Jill went u
p the hill
2:to fetch a pail of w
ater.
3:Jill fell down and b
roke her arm,
4:and Jack came tumbli
ng after.
```

The lines of the file are displayed in sequence with the line number in front of each line.

The C (Copy) and M (Move) Commands

The **C** (copy) command permits you to copy a line or series of lines, from one place in the file to another place in the file. You can also copy all or part of another file into your edit file. **C** always inserts the copied text before the current line. This command is very handy if you have written something you need to use repeatedly.

[beginning line number [ending line number]] C [filename]

The `M` (move) command acts just like the `C` command except the text is deleted in the original location.

[beginning line number [ending line number]] M [filename]

Default values: Edit file: *beginning line number* = current line
 ending line number = beginning line number

 Other file: *beginning line number* = line 1 of other file
 ending line number = last line of other file

If no *filename* is specified, the indicated lines come from the edit file. If a *filename* is specified, the indicated lines come from the specified file. You can't copy or move a block of text that includes the current line, unless the current line is the first or last line of the block of text. If the current line is *not* the first or last line, you will get the Invalid Parm error message.

The `Working...` message is displayed when you copy or move text.

Here are some examples of the `□` and `▮` commands:

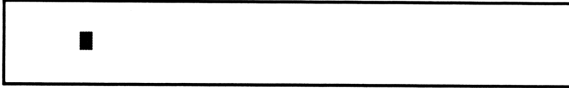
C	Duplicate the current line.
20 C	Copy line 20 and insert it before the current line.
3 9 M	Move lines 3 through 9 from within the edit file, insert them before the current line, and delete the original lines 3 through 9.
C CAT	Copy the file CAT and insert the lines before the current line.
20 C FROG	Copy lines 20 through the last line of the file FROG and insert the lines before the current line in the edit file.

Example: Begin a new file called MEMO.

Input/Result

```
EDTEXT MEMO [END LINE]
T [END LINE]
```

Execute the Text Editor, creating a new file MEMO, then enter text mode.



You are now ready to enter text in your file MEMO.

Enter the following text for MEMO. Don't forget to end each line by pressing the [END LINE] key.

```
John-
The following short nursery rhyme can be used
for demonstrating the editing of text files
using the TEXT EDITOR. It is a familiar
rhyme and easy to enter into a file.
[END LINE]
```

The [END LINE] was entered for line 6 because we want a blank line between the memo and the rhyme. *The C command always inserts the copied text before the current line.* Therefore, the current line in this example must be line 7 if the new text is to be inserted beginning with the new line 7.

Press the [RUN] key to return to the Command Prompt.

Now copy the file SAMPLE to the end of the MEMO file: C SAMPLE [END LINE].

The nursery rhyme is added to the end of your MEMO file. You can check your MEMO file by using the L (list) command.

The I (Insert) Command

The I (insert) command permits you to add a line or a series of lines into the middle of a file. You used this command in section 1 when you added a line to your SAMPLE file. Remember that when you use I the new lines are inserted *before* the current line.

```
[line number] I
```

Default value: *line number* = current line

When you enter the insert mode, the line specified by *line number* appears in the display. If you are inserting at the end of the file, Eof will be in the display. You will also see that flag 1 is on, indicating that you are in insert mode. The cursor will not appear until you begin entering new characters on the line, or moving through the file by pressing `↑` or `↓`. To exit from the insert mode, press `RUN` or `ATTN`.

Example: Suppose that your edit file is SAMPLE, containing the modified nursery rhyme.

```
1:Jack and Jill went up the hill
2:to fetch a pail of water.
3:Jill fell down and broke her arm,
4:and Jack came tumbling after.
```

Suppose that you want to enter some lines at the beginning of the file to prepare it for printing. (Formatting commands are described in section 3, page 39.)

Input/Result

11 `END LINE`

Make line 1 the current line and enter the insert mode.

```
Jack and Jill went up '
```

The current line is now line 1. Notice the flag 1 annunciator is visible in the display indicating insert mode.

At this point, you can use `↑` and `↓` to move through the file. If you do so, the current line will change, so make sure that you move back to line 1 before entering the new line.

Input/Result

```
^MA 20 60 ^CE Nursery Rhyme
END LINE
```

When the F command is executed, this line sets the margins and centers the title.

```
Jack and Jill went up
```

The new line has been inserted at the start of the file. The current line remains at Jack and Jill... and is now line 2. The new line is line 1.

```
^SK 2 END LINE
```

This line tells the printer to skip two lines.

```
Jack and Jill went up
```

The new line has been inserted before the current line Jack and Jill..., which is now line 3. The new line is line 2.

Press **RUN** or **ATTN** to return to the Command Prompt.

Input/Result

RUN

```
Line 3, Cmd:■
```

1#LN **END LINE**

Now list the file to see the new lines.

```
1: ^MA 20 60 ^CE Nurser
y Rhyme
2: ^SK 2
3: Jack and Jill went u
p the hill
4: to fetch a pail of w
ater.
5: Jill fell down and b
roke her arm,
6: and Jack came tumbli
ng after.
```

The **D** (Delete) Command

The **D** (delete) command deletes lines from the edit file. If you wish, you can place the deleted lines into a new file, or, using the **+** option, append the lines to an existing file. If the specified file already exists and you do not use the **+** option, the error message **File Exists: filename** will appear. You can not purge a file while you are in the Text Editor, but you can delete all of the text and leave an empty file. Refer to your *HP-71 Owner's Manual*, section 6, for instructions on how to purge a file.

```
[beginning line number [ending line number]] D [filename [+]]
```

Default values: *beginning line number* = current line
ending line number = beginning line number

When you execute **D** with line number parameters specifying more than one line, the message **OK to Delete? Y/N:** will come up. This is to prevent accidental deletions, as you must answer **Y** before the Text Editor will complete the deletion. If you answer **N**, the Command Prompt returns, and the lines are not deleted.

The `Working . . .` message is displayed when you use `□`.

The following examples show some uses of the `□` command:

<code>D</code>	Delete the current line.
<code>12 32D</code>	Delete lines 12 through 32.
<code>4 9 D CACHE</code>	Delete lines 4 through 9 and store them in a new file called <code>CACHE</code> .
<code>2 21D ARCHV+</code>	Delete lines 2 through 21 and append them to the end of a file called <code>ARCHV</code> .

The S (Search) and R (Replace) Commands

The **S** (search) and **R** (replace) commands allow you to search through a file for a certain string of characters. If the string is found, **S** returns the Command Prompt; the current line number is that line where the string was found.

R finds all occurrences of the string within the line parameters and replaces the old string with a new string. The Command Prompt returns, and the current line is that line containing the last occurrence of the string.

If the search string is not found, the message `Not Found` is displayed and the current line number and Command Prompt return to the display.

[beginning line number [ending line number]][?] S/string[/]

[beginning line number [ending line number]][?] R/string1/string2[/]

Default values:

<i>beginning line number:</i>	search = current line plus 1 replace = current line
<i>ending line number:</i>	search = last line replace = beginning line number

These commands search the edit file between the line number parameters for the string indicated between the slashes (/). These slashes, sometimes called *delimiters*, mark the boundaries of a string. Any character (except a blank space) may be used as the delimiter. The first non-blank character after the command S or R will be used as the delimiter. The last delimiter is optional (as indicated by the brackets), unless another command follows this command. (Refer to “Concatenating Commands” on page 36.)

The `?` in the command string indicates the question mark option. When this option is used, the program finds the string, then shows you the location of the string and waits for you to decide if this is the occurrence you want. The question mark option is described below.

`S` and `R` distinguish between uppercase and lowercase letters. For example, a search for the string `jack` will not find the string `Jack`.

The `Working...` message is displayed when `R` is used.

The following examples show some `S` commands and `R` commands with parameters:

<code>S/Jack</code>	Search from the current line plus one for the first occurrence of the string <code>Jack</code> . If found, designate that line as the current line and return to the Command Prompt.
<code>3 7 S/Jill</code>	Search from line 3 through line 7 for the string <code>Jill</code> . If found, designate that line as the current line and return to the Command Prompt.
<code>R/cat/dog/</code>	Replace <code>cat</code> with <code>dog</code> at all occurrences of <code>cat</code> in the current line.
<code>4 7R/cat/dog</code>	Replace <code>cat</code> with <code>dog</code> at all occurrences in lines 4 through 7.
<code>.#R*3/4*3/8</code>	Replace <code>3/4</code> with <code>3/8</code> at all occurrences of <code>3/4</code> from the current line to the end of the file. The character <code>*</code> is used as the delimiter so that slashes may occur in the strings.
<code>1#R/meet//</code>	Delete all occurrences of the string <code>meet</code> from line one to the end of the file, by replacing it with the null string.

The question mark option allows you to check the location of the found string so you can determine if it is the occurrence you want. When you include `?` in the command, you have three options when your string is located: ☐ `Y` (yes), ☐ `N` (no) or ☐ `Q` (quit).

Using `S`, your options are:

- Press ☐ `Y` to make that line the current line, in which case you will see the line number and the Command Prompt.
- Press ☐ `N` to make the program search for another occurrence of the string.
- Press ☐ `Q` to quit the search and return to where you were before you executed `S`.

Using `R`, the options are:

- Press ☐ `Y` to replace *string1* with *string2* in that line and search for the next occurrence of *string1* within the line parameters specified. If there is another occurrence, the program will find it.
- Press ☐ `N` if you do not wish to replace *string1* with *string2* at that place in your file. The program will continue to search for the next occurrence of *string1* within the line numbers specified.
- Press ☐ `Q` to quit the replacement search and return to the last replacement made or to where you were before you executed `R`, if no replacements were made.

If you press any other key, except `[ATTN]`, the display will show `Y/N/Q ?` to indicate that only `[Y]`, `[N]` or `[Q]` are permitted as responses at this point. If you press `[ATTN]`, the Command Prompt will return to the display.

When you use the question mark option, the display shows the following information when the string is found:

- the line number
- the column number in which the first letter of the string occurs
- that portion of the line beginning with the string, after scrolling through the line
- a question mark (?) indicating the option

Examples:

```
.?S/string
```

Search from the current line for `string` and, if found, display the line number, column number and part of the text in that line, and allow the question mark option.

```
?R/string1/string2
```

Search for `string1` starting in the current line but stop when `string1` is found to display the line number, column number and part of the text in that line, and permit the user to select `[Y]`, `[N]` or `[Q]`.

Example: Suppose that your edit file is `SAMPLE`. Suppose that line 6 is the current line. Search for the word `fell`. Since the current line is at the end of the file, use the *beginning line number* parameter to search from line 1 in the file.

Input/Result

```
Line 6, Cmd:■
```

```
1?S/fell [END LINE]
```

```
5:6\fell down and br/?
```

`S` searches for the first occurrence of the word `fell` starting on line 1. The following information appears in the display when the string is found:

- The line number (5).
- Line number/column number separator (:).
- The column number in which the first letter of `fell` occurs (6).

- A backslash delimiter (\).
- The portion of the line beginning with the string (fell down and br).
- A slash delimiter (/).
- ? indicates the question mark option and that a ☐Y/☐N/☐Q response is needed.

Select the ☐Y option.

☐Y

Line 5, Cmd:■

Line 5 containing the word fell becomes the current line.

If you had not used the question mark option in the original S command, the Text Editor would have found the first occurrence of the word fell in the text then displayed the line number followed by the Command Prompt. To find other occurrences of the string in the text, the command would have to be reentered.

If the program does not find the search string, the display shows the message Not Found and the current line number and Command Prompt return to the display.

Now replace the word pail with the word bucket.

Input/Result

Line 5, Cmd:■

1#R/pail/bucket ☐END LINE

Line 4, Cmd:■

The word pail is replaced by the word bucket and line 4 becomes the current line.

Suppose now that you wish to replace the name Jill with Judy in the second occurrence, but not in the first occurrence. Also suppose that you do not know the lines in which the name Jill appears. Use the question mark option.

Input/Result

```
1#?R/Jill/Judy 
```

```
3:10\Judy went up th/?
```

The display shows the line number and the column number in which the first occurrence of `Jill` appears. The backslash and the slash enclose part of the line beginning with the search string. `Judy` has replaced `Jill` in the display, but the line has not been changed in the file. The program awaits your selection of , , or .

Select the option to *not* replace on this line and search for the next occurrence.

```
5:1\Judy fell down a/?
```

The display shows the line number and the column number of the first character of the string `Jill`. The backslash and the slash enclose a portion of the line of text with `Judy` replacing `Jill`. The replacement has not yet been made in the file.

Select the option to replace `Jill` with `Judy` in that line.

```
Line 5: Cmd:█
```

The replacement takes place and the search continues for the next occurrence of `Jill`. Finding none, the Command Prompt returns, at the line where the replacement occurred.

Using `R`, it is possible to create lines longer than 96 characters in your text files. When you do replace into a line that is longer than 96 characters, using the `?` option you will be able to scroll through only the 96 character substring that contains that search string, rather than the whole line. If you try to go into text mode on this line, you can't—you will get the `Line Too Long` message.

Special Patterns

A feature of the `S` and `R` commands is the availability of five characters that have special meaning when used in patterns. Using these characters in a search or replace string tells the Text Editor to look, for example, only for those occurrences of the string at the beginning of the line, or at the end of the line, or to allow *any* pattern between two specified patterns. The five characters that can be used in this special way are `.`, `@`, `&`, `^` and `$`.

The backslash (`\`) character is used in the command like a “switch” that *starts* and *stops* this feature. (The backslash has been assigned to `f()` by the Text Editor program.) The first occurrence of the backslash turns on the feature, and the five characters take on their special meanings. The next occurrence of the backslash turns this feature off. It is also turned off by the delimiter ending the string.

The five characters, their meanings and some examples of their uses are described in the following paragraphs:

- The period (`.`) is a “wild card” character. The Text Editor searches for the specified string, but any character can be in those positions in the string where you put a period.

Example: `1#R/ABC\...\Recheck ID#` will replace the occurrences of `ABC` followed by any three characters, such as `ABC999`, or `ABCzyz`, or `ABCyz`, with the string `Recheck ID#`.

`1#R/ABC\...\Recheck ID#` is identical to the example above. The second backslash is not needed because the `/` at the end of *string1* stops the special feature. The ending slash is optional for *string2*.

Example: `1#R/\...DEF\...ABCDEF` will replace a string consisting of any first three characters followed by `DEF` followed by any two additional characters. The replacement string is `ABCDEF`. The second backslash is not needed because the end of *string1* stops the special feature.

- The commercial “at” symbol (`@`) indicates that any number of characters between the beginning of a string and the end of a string on the same line are “wild cards”—that is, there can be any number of characters—you don’t have to specify how many characters, or what they are. Because the program starts searching for the end of the string at the end of the line, the longest match is found.

Example: `1#R/ABC@CDE/Recheck ID#` will replace the occurrences of a string beginning with `ABC` and ending with `CDE` on the same line, such as `ABC123CDE`, `ABCCDE`, or `ABC12 zzzCDE`, with the string `Recheck ID#`.

- The ampersand (`&`) is used only with the `R` command and can only appear in *string2*. Every place that the ampersand appears in *string2*, it is replaced by the string matched by *string1*. *String2* is then inserted into your file.

Example: `1#R/\AB.\&DEF` searches for the occurrences of the string `ABwildcard`, and appends the string `DEF` to it. If `ABC` is found, the new string will be `ABCDEF`.

- The circumflex (^) is used to find a string only when that string appears at the *beginning* of a line. If the string appears anywhere else in the line, it will be ignored. The circumflex has this special meaning only when it appears as the first character of the string. Anywhere else in the string, ^ will have its normal meaning.

Example: `1#R/\^ABC/CDE/` will search for the string ABC only at the beginning of a line. If ABC appears anywhere else in the line, a match will not be made.

Example: `1#R/\^3.14159/2.71828` will search for 3.14159 at the beginning of a line and replace it with 2.71828. In this case, the second backslash is necessary. If not included, the decimal point in 3.14159 would be interpreted as a special period. A slash at the end of *string2* is optional.

- The dollar sign (\$) in *string1* tells the Text Editor to search for *string1* only at the *end* of a line. The dollar sign character must appear at the end of *string1*. When it appears anywhere else in the string, it has its normal meaning.

Example: `1#R/ABC\$/CDE` will search for the string ABC only at the *end* of a line. If ABC appears anywhere else in the line, it will be ignored. A second backslash is not needed after the \$ because the dollar sign is at the end of *string1*.

The backslash can enclose a mixed series of special characters. If the sequence cannot be understood, the Text Editor returns the message `Invalid Parm:` then the command line with the command string returns to the display.

Example: `1#R/\^...ABC/Check this line/` will search the beginning of each line for any three characters followed by the string ABC. The command will replace the six characters with the string `Check this line` only when the six characters appear at the beginning of a line. The second backslash is unnecessary because there are no other special characters in *string1*.

Example: Suppose you have loaded a text file from the HP-75 into your HP-71. Now you want to get rid of the four-digit line numbers that the HP-75 put at the beginning of every line. `1#R/\^....//` tells your HP-71 to search, from line 1 to the end of the file, for any four characters at the beginning of the line, and replace them with nothing (delete them).

Sometimes, your string may contain a backslash character as part of the text. In this case, you don't want S and R to see the backslash as a switch. The solution is to use two sequential backslashes. The Text Editor will interpret `\\` as a single backslash character, not as a switch.

The F (Format) Command

The F (format) command produces a printed document from the edit file. Text formatting commands embedded in your file determine how your formatted, printed document will look.

F [*number of copies*] [*fixed space-character*]

Default values: *Number of copies* = 1
 Fixed-space character = ~

Refer to section 3, “Text Formatting Commands,” for a description of the text formatting commands, and to page 43 for information about the fixed-space character. The default character is the tilde, ~, which has been assigned to **f** **=** by the Text Editor.

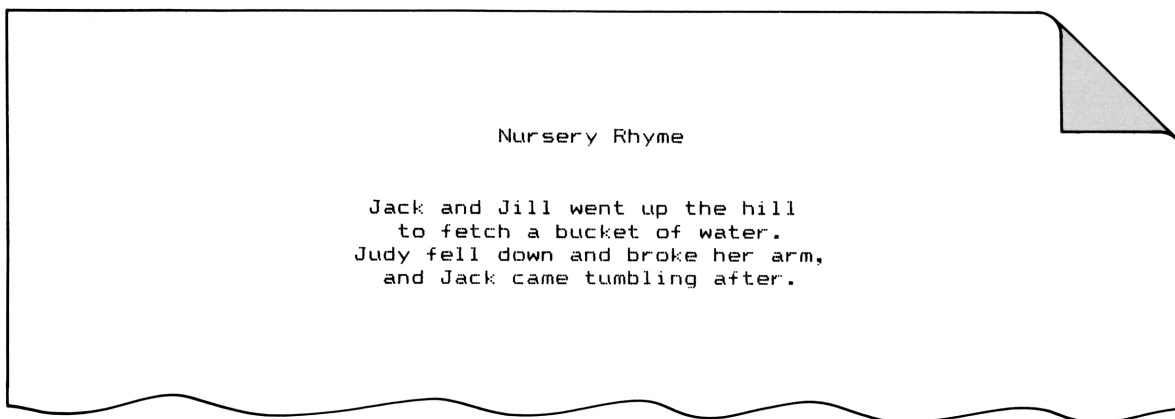
The message `Working...` is displayed during the formatting and printing operations.

Example: Format and print `SAMPLE`. Make sure the file contains the text formatting commands added in “The I (Insert) Command” on page 26.

Input/Result

F **END LINE**

The Text Editor will format, then print `SAMPLE`.



Concatenating Commands

More than one Text Editor command can be put into a command string if a semi-colon (;) is used as a separator between commands.

The Text Editor processes these commands in order. If there is an error in one of the commands, the appropriate error message appears, followed by that portion of the original command string from the error to the end. The prior commands have already been completed. If you now correct the command and press **END LINE**, the operations will continue.

Example: 1PN;1;.?S/Frogs are green

This command string will perform the following operations:

- Print the edit file, beginning at line 1, with line numbers (refer to L (list) and P (print), page 22).
- Make line 1 of the edit file the current line.
- Search for the string `Frogs are green` in the edit file using the question mark option (refer to the S (search) command on page 28).

If a command is concatenated to an S (search) or R (replace) command, the last delimiter in the command is *required* rather than optional. If the last delimiter is not used, S or R will interpret the concatenated command as a part of the search or replacement string.

Example: 4 10 S/Frogs are green;/I

This command string will perform the following operations:

- Search for the string `Frogs are green` in lines 4 through 10.
- Switch to the insert mode at the line where the string is found, or if the string is not found, insert mode will be in effect at the current line.

If the second slash was not present in the command string, the program would search for the string `Frogs are green;I`.

The H (Help) Command

The H (help) command displays the syntax of the Text Editor commands in alphabetical order. You can move through the commands, up or down, by pressing **↑**, **↓**, **G↑** or **G↓**. To exit the help sequence at any point, press **RUN** or **ATTN**, and the Command Prompt will return to the display.

H

Example: Suppose that you are editing a text file and want to see the syntax of the `D` (delete) command.

Input/Result

```
Line 12, Cmd:
```

H `END LINE`

Select the H command.

```
b[e] C [<file>]
```

`C` (copy) command syntax is displayed.



```
[b[e]] D [<file>][+]
```

`D` (delete) command syntax is displayed.

`RUN`

Press `RUN` or `ATTN` to terminate help.

```
Line 12, Cmd:
```


Text Formatting Commands

This section describes how to use the text formatting commands in general, then discusses the specific meaning and use of each command. For an alphabetical listing of the commands, refer to the “Command Summary and Index” at the back of the manual. Remember that when you are ready to print out your text file, press F when the Command Prompt is in the display to run the formatting routine.

General Syntax for Commands

Text formatting commands are put into a text file either while the text file is being written or afterwards. All the commands are two-letter mnemonics (such as `^MA` for *margin*), which you can type in upper or lower case letters. You can string together more than one command on one line, and you can follow command(s) with text, but *any commands on a line must appear before any text on the same line*. Any command appearing after text will be interpreted as text.

The single line

```
^sk ^pa This is a demonstration of the
```

is equivalent to the following three lines:

```
^sk
^pa
This is a demonstration of the
```

The Command Character (^)

Every individual text formatting command in a text file must be preceded by the *command character*, the circumflex (^). It is the command character that signals the presence of a command to the formatting routine.

Parameters

Each command can be followed by one or more parameters, such as numbers or filenames. If a parameter is *optional*, you don't have to supply that parameter. If you don't supply an optional parameter, then the computer uses a *default value* for that parameter.*

If there are two optional parameters, you can omit them both or omit just the second one, but you cannot omit just the first one.

```
^ma
```

Uses the default parameters 13 (left margin) and 72 (right margin).

```
^ma 8
```

Uses the default parameter for the right margin, so the margins are 8 and 72.

```
^ma 8 68
```

Sets margins to columns 8 and 68.

Separating Parameters. If you give more than one parameter for a command, the parameters *must* be separated from each other by one or more *spaces*. However, you do not need a space between a command and a following parameter. Commas are not allowed as separators in text formatting commands.

```
^ma1 9
```

Sets margins to columns 1 (left) and 9 (right).

```
^ma19
```

Sets margins to 19 (left) and, by default, 72 (right).

Default Values. A default value takes effect until some other value is explicitly given. The default values for all the conditions and commands are re-established each time you use the F command. They are then altered by specific commands in your text file. Therefore, any conditions that you set up in one text file (like margins, spacing, page numbering, and so on) *will not have any effect on any other file*.

Numeric Values. Numeric parameters must be integers. The range for a parameter can depend on other conditions; for instance, it is not possible to set a paragraph to indent further than the right margin!

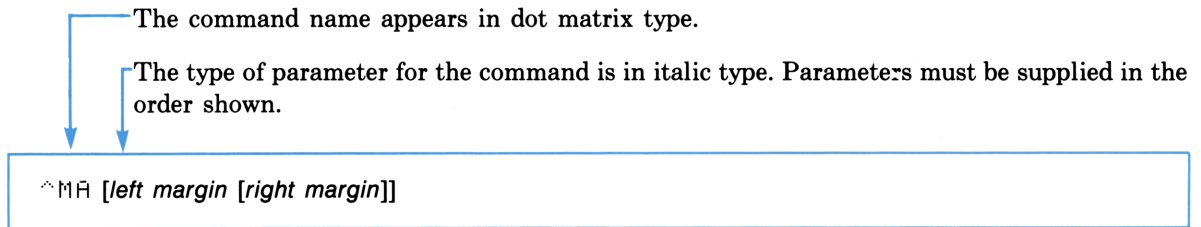
* A *default* condition or parameter is one that is in effect “by default” unless you specifically change or redefine that condition.

Text Input and Output

The formatting routine ignores any trailing (ending) spaces in a line of text from your text file. It automatically adds a space between the last word from one line and the first word from the next line in the file. It will not divide or hyphenate words between lines.

Typing Conventions in This Section

Each command described in this section includes a syntax box, which gives the following information:



If the parameters are optional, they are enclosed in brackets. Nested brackets, as in this example, indicate a hierarchy: the value for the left margin can be given without the value for the right margin, but the right margin cannot be given without the left margin.

Note that items for you to type in exactly as shown are printed in DOT MATRIX type, while items that are merely descriptions are printed in *italic* type. Specific keys on the HP-71 are shown in key boxes, such as `END LINE`.

Text Formatting Modes

There are four *formatting modes* for formatting lines of text. They are set by the commands `^FI` (*Fill*), `^JU` (*Justify*), `^CO` (*Copy*), and `^CE` (*Center*). Both Fill and Justify are filling operations, while both Copy and Center are copying operations. A *page-test parameter* is optional with a text mode command.

Filling Lines and Justifying Lines (`^FI`, `^JU`)

When in a text-filling mode, the formatting routine *fills up* every line it prints out with as many whole words as will fit. To produce one line of formatted, printed text, as much text as is necessary to fill the line is used from your text file, normally ignoring any extra spaces in the text file. This is why it doesn't matter how much or how little you write on a line in your text file—when the formatting routine runs, it fills up each line as much as it can, given its margins. The left margin is justified.

The two *text-filling modes* are Fill mode and Justify mode.

```
^F I [page-test parameter]
```

```
^J J [page-test parameter]
```

- *Fill mode* (^F I) is the *default* text mode for the formatting routine. Every time you execute the F command, Fill mode is automatically set. Fill mode operates as described above.
- *Justify mode* (^J J) justifies the right margins by adding spaces between words. Otherwise, it operates like Fill mode.

In either of these modes, a *single* space following a sentence terminator (., :, ?, !) is expanded to two spaces.

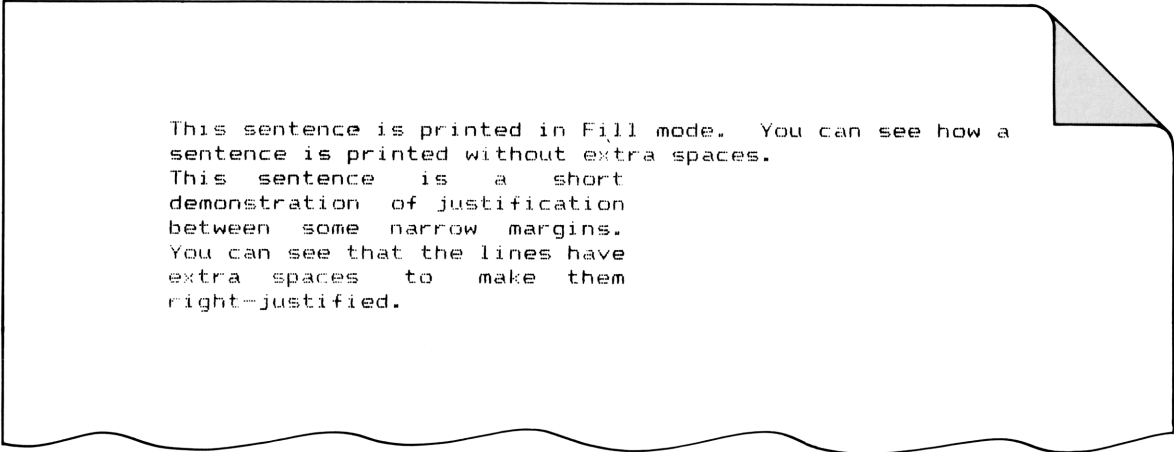
If you specify a page-test parameter, these commands will also check how many lines remain on the page being printed. The page-test parameter checks for a minimum number of lines. Page testing is explained below (page 46) under “Testing for the End of the Page.”

Example: The following short text file shows the effects and differences of Fill mode and Justify mode. To create this file, type EDTEXT TEST,T;F;E (T for text mode, F for format and E for exit), then enter the text.

```
This sentence
is
printed in Fill
mode.
You can see how a sentence is
printed without extra spaces.
^ma 13 43 ^ju
This sentence is a short
demonstration of justification
between some narrow margins.
You can see that the lines
have extra spaces to make them
right-justified.
```

RUN

The formatted, printed output is:



```

This sentence is printed in Fill mode.  You can see how a
sentence is printed without extra spaces.
This sentence is a short
demonstration of justification
between some narrow margins.
You can see that the lines have
extra spaces to make them
right-justified.
  
```

Keeping Words Together (Fixed Space). If you want to be sure that a particular pair of words won't be split between lines during printing, or you don't want the justification procedure to add extra spaces between them, put a *fixed-space character* instead of a space between the two words. The fixed-space character is the tilde, ~. The Text Editor program assigns ~ to **f** **=**. You can define it to be another character by using the optional fixed-space character parameter when you enter the F command.

For instance, if you want to be sure that the phrase "HP-71 Computer" is printed all on one line and (in Justification mode) without any extra spaces between the two words, type it into your text file as HP-71~Computer. The ~ character will *not* be printed; it will appear as a space.

Duration. When Fill mode or Justify mode is set, it lasts for the duration of that particular formatting operation, unless you change it. (Fill mode is automatically set each time F runs.) However, these two text-filling modes are *ended* by the text-copying modes, copy (^CQ) and center (^CE). After copying or centering ends (as explained below under "Copying and Centering Lines") F sets Fill mode.

Breaking. There are several commands that involve a *break* in the output of the current printing line. Printing then resumes on a *new* line. For instance, when a command to skip lines (^SK) occurs, the formatting routine *stops adding words to the current line of print*, performs the skip, and resumes printing the subsequent text on a new printed line.

As an example, in Justification mode, the last line of a paragraph is not justified, because the output for that line is broken by a subsequent ^PA, ^SK, or ^CO command (as examples). Also, the very last line from a file is not justified.

The commands that cause a break in output and start new lines of print are:

^AD <i>Advance Page</i>	^FI <i>Fill</i>	^PA <i>Paragraph</i>
^CE <i>Center</i>	^JU <i>Justify</i>	^SK <i>Skip</i>
^CO <i>Copy</i>	^MA <i>Margin</i>	

You can use ^SK 0 to cause printing to occur on the next line.

Copying Lines and Centering Lines (^CO, ^CE)

There are times when you won't want the formatting routine to "fill up" your lines, but instead will want it to print out a line of text exactly as it appears in the text file. This is called *copying* a line, and is done by setting ^CO or ^CE.

The two *text-copying modes* are Copy mode and Center mode.

^CO [*page-test parameter*]

^CE [*page-test parameter*]

These commands also test for the end of the page if you specify a page-test parameter. Refer to "Test-
ing for the End of the Page" on page 46.

- *Copy mode* (^CO) causes the subsequent text to be printed line-by-line as it appears in the text file. Spaces between words are preserved. *Leading spaces are also preserved* though trailing spaces are not.
- *Center mode* (^CE) copies (just like ^CO) and centers subsequent text, line by line. In addition, *both leading and trailing spaces are ignored*.

Duration. Copy or Center mode remains in effect for subsequent lines *until* the occurrence of an ^F I, ^J U, ^C O, ^C E, ^P A, or ^T A command. Following a ^P A (*paragraph*) or ^T A (*tab*) command, Fill mode is set. *Copy and Center modes are not restored after the execution of ^P A or ^T A.*

Example:

```
These words  appear
in
```

```
    Fill  mode.
```

```
^sk 1 ^coThese words  appear
in
```

```
    Copy  mode.
```

```
^sk 1 ^ce Center just this line.
```

```
^fi This is the next line in
Fill mode.
```

Fill mode is set.

Skips one line and sets Copy mode.

Sets Center mode.

Sets Fill mode.

RUN

The formatted, printed output is:

```
These words appear in Fill mode.
```

```
These words  appear
in
    Copy mode.
```

```

                Center just this line.
This is the next line in Fill mode.
```

Line Length. If a line in the text file is too long to be printed within the given margins, only as much of the line as fits on one line will be printed. To indicate that characters were truncated, the last two characters printed on that line will be question marks, and the first three characters on the next line will be asterisks.

Testing for the End of the Page

If you want to ensure that a section of text (like a table or a short paragraph) will not be split between two pages of output, you can include an optional *page-test parameter* with any of the text mode commands. The page-test parameter represents the number of lines that you want to keep together. This activates the page-testing function to check that there is enough room for that many lines before the end of the page. If there is enough room (at the time of printing), then the formatting routine continues to print out text. If there is *not* enough room, then the page advances before more text is printed.

In order for page testing to function properly, the page-length setting must be correct (see `^PL`, page 50), and the formatting job must begin with the printer's printhead aligned with the top of a page. (It should be set on the first line after a perforation.) The following example uses `^FI 4` to check that there are at least four lines left on the page before printing out a short paragraph. In this example, Fill mode is set initially, so `^FI 4` does not change the current text mode.

```
^pa Start of paragraph one.
This paragraph continues for
several lines.
```

```
:
```

```
End of paragraph one.
^fi 4 ^pa Begin a new paragraph
with this sentence.
```

Fill mode on.

Checks for the end of the page within four lines. If there are less than four lines available, then the page advances before the next paragraph is printed.

Indenting Lines

Two other commands that affect the format of a line are *paragraph* (`^PA`) and *tab to column* (`^TA`). After the formatting routine makes the indentation for a paragraph or tab, Fill or Justify mode is set, whichever was last in effect. *Either of these commands will cancel Copy or Center mode*, and set Fill mode.

Starting a New Paragraph (`^PA`)

```
^PA [number of spaces to indent]
```

Default value: *number of spaces* = 5 or the last value used.

This command starts a new line of printed text, indenting it the given number of spaces. If no number is given, then the indentation is the same as the last time `^PA` was used, so you don't have to specify the indentation except to change it. If you don't ever specify an indentation, five spaces are used.

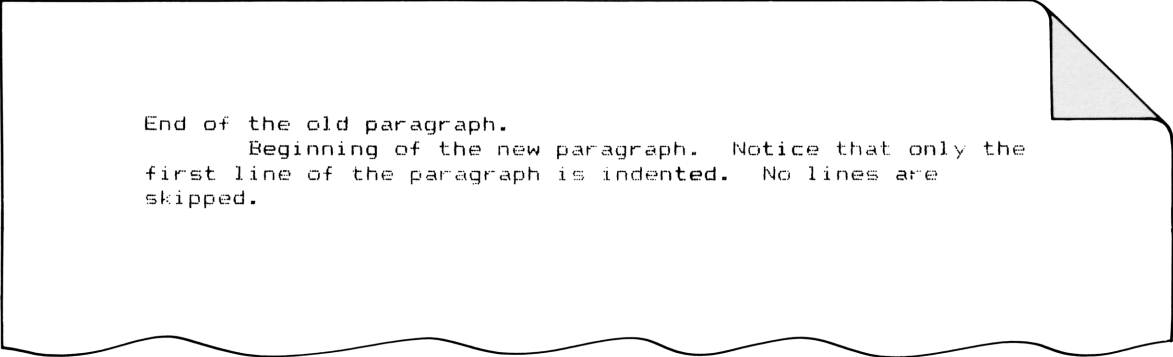
`^PA` indents from the current left margin, so you can specify a *negative* indentation to start the line to the *left* of the current margin. If, for example, the left margin is set to 8 and you use a `^PA -5`, then the first character of the next line will start in column 3. `^PA 0` does not indent but skips a line instead. (`^PA` with other values does *not* skip a line.)

`^PA` includes an implicit page test for two lines. This avoids having just the first line of a paragraph appear at the bottom of a page.

Example:

```
End of the old paragraph.
^pa 7 Beginning of the new paragraph.
Notice that only the first line of the
paragraph is indented. No lines are
skipped.
```

The printed, formatted output is:



```
End of the old paragraph.
    Beginning of the new paragraph. Notice that only the
    first line of the paragraph is indented. No lines are
    skipped.
```

Indenting by Tab to a Column (`^TA`)

```
^TA column number
```

The `^TA` command indents the printed text to the given column position on the same line, if possible. Regardless of the previous mode, the line being tabbed is never justified.

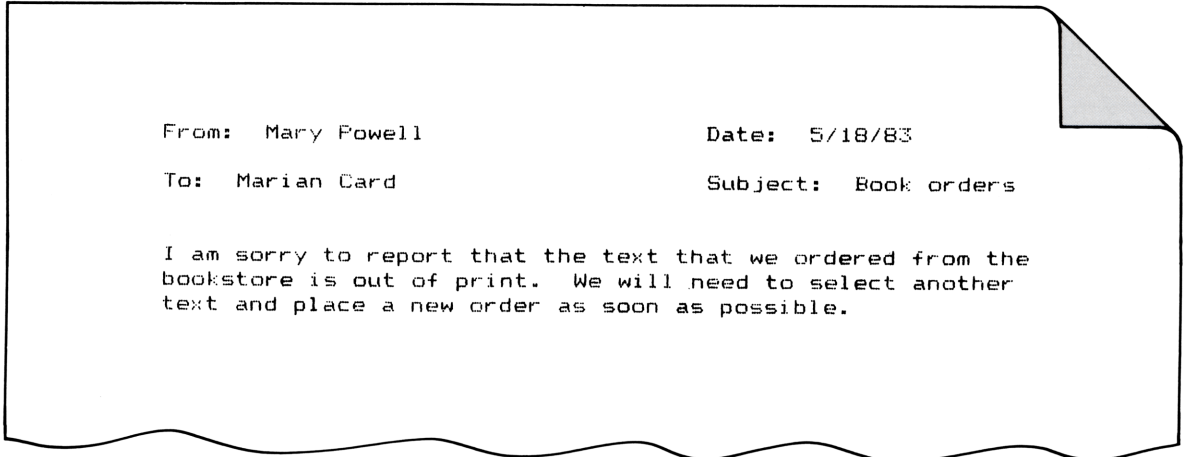
Example:

```

From: Marian Card
^ta 50 Date: 5/18/83
^sk To: Mary Powell
^ta 50 Subject: Book orders
^sk 2 I am sorry to report that the text
that we ordered from the bookstore is
out of print. We will need to select
another text and place a new order
as soon as possible.

```

The formatted, printed output is:



```

From:  Mary Powell           Date:  5/18/83
To:    Marian Card          Subject: Book orders

I am sorry to report that the text that we ordered from the
bookstore is out of print.  We will need to select another
text and place a new order as soon as possible.

```

If the given column number is less than the current column position in the printed line, the text output will begin in the given column on the *next* printed line.

Formatting a Page

The text formatting commands covered under this topic are concerned with formatting the output of a page as a whole, rather than individual lines of text. These commands are `^MA` (*set margins*), `^PL` (*page length*), `^SP` (*set line spacing*), `^SK` (*skip lines*), `^PH` (*page numbering*), and `^AD` (*advance page*).

Setting Margins (^MA)

```
^MA [left margin [right margin] ]
```

Default values: *left margin* = 13
 right margin = 72

Each time you run the formatting routine, the left and right margins for the printed page are automatically set to columns 13 and 72. These same default values are set by the command ^MA without any parameters. Or, if only the left margin is set, the right one is automatically 72.

Example:

```
^ju This text is printed within
the default margins of 13 and 72.
^ma 17 65 ^sk
You can highlight a block of text by
setting margins to 17 and 65, as in
this example. When you have completed
the text that you wish to highlight,
you can use the MA command with no
parameters to reset the margins.
^ma ^sk
This is the last sentence. It is
printed within the default margins
again.
```

Resets margins to 17 and 65.

Resets margins to 13 and 72.

The printed, formatted text is:

This text is printed within the default margins of 13 and 72.

You can highlight a block of text by setting margins to 17 and 65, as in this example. When you have completed the text that you wish to highlight, you can use the MA command with no parameters to reset the margins.

This is the last sentence. It is printed within the default margins again.

The margin numbers refer to column numbers on the page. Although 80 columns is the normal width of a page, you can specify a right margin up to 132. The left margin can be as low as 1.

Page Length (^PL)

^PL [*page length* [*top margin* [*bottom margin*]]]

Default values: *page length* = 66
top margin = 6
bottom margin = 6

The ^PL command tells the formatting routine how many lines of print are possible per page (the total page length), and how many lines to leave as margins on the top and bottom of the page. The usual, 8½"×11" sheet is 66 lines long for an HP 82905B printer. If you don't set the top and bottom margins, the first and last six lines on each page are automatically skipped.

The specifications for page length are necessary for the proper operation of end-of-page testing, page numbering, and paper advancement while a file is being formatted and printed. For the page length to be properly counted, the formatting job must begin with the printer printhead aligned with the top of a sheet of paper.* (You should set the printhead at the first line after a perforation.)

If you are using a sheet-feed printer, for which you manually insert each sheet of paper, use a *negative* number for the page length parameter (for example, -66). The HP-71 will then display this message each time a new sheet is needed:

```
Insert Page...
```

You can then insert a new sheet of paper, align the printhead on the first possible line of the paper, and press **END LINE** to continue.

If your printer already automatically skips a certain number of lines for top and bottom margins, then the formatting routine will skip *additional lines* according to the margins that are set. Therefore, you should know what your printer does for top and bottom margins. (Check the printer manual.)

Examples: Suppose you have a sheet-feed printer, printing 60 lines per page, that begins printing on the fourth line of each page. You want one-inch margins at the top and bottom of the page, and the printer prints six lines per inch. Use this page-length command:

```
^pl -60 3 6
```

Since this printer already skips three lines at the top of a page, your top margin parameter should be 3 (not 6), creating a total margin of six lines.

Printers With Perforation Skipping. Some types of thermal printers that use roll or fanfold paper (like the HP 2671A printer) have a perforation-skip feature. This causes the printer to advance past the perforations in the paper, skipping the last and first three lines on each sheet of paper. These printers also usually print 60 lines per page (by default). To obtain a one-inch (six-line) margin at the top and bottom of each page, use this page-length command:

```
^pl 60 3 3
```

* The `^PL` command should be set at the beginning of the text file so that it will be set when the printer is at the beginning of the page. Otherwise, `^PL` could cause the printer to advance to the next page.

Roll-Fed Printers. You can avoid page-length considerations entirely by specifying the page length as `^PL 0`. This is desirable, for instance, if you are using a roll-fed printer, such as an HP 82162A printer.

`^pl 0`

Do not specify top and bottom margins in this case.

Line Spacing (`^SP`)

`^SP [number of lines]`

Default value: *number of lines* = 1

The `^SP` command sets the spacing between lines of printed text output. You can set a value from 1 to 5, with 1 for single-spacing, 2 for double-spacing, and so on.

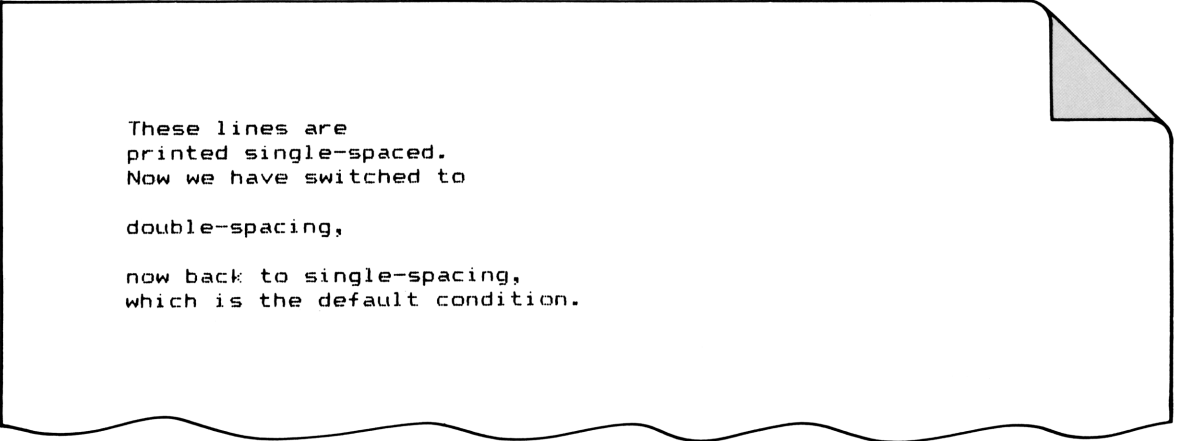
`^SP` does not cause a break in printing, that is, it does not cause the printing to stop on one line and start again on a new one.

Example:

```
^coThese lines are
printed single-spaced.
^sp 2Now we have switched
to double-spacing,
^spnow back to single-spacing,
which is the default condition.
```

Since this is in Copy mode, the formatting routine preserves any spaces between a command and the text.

The formatted, printed output is:



```
These lines are  
printed single-spaced.  
Now we have switched to  
  
double-spacing,  
  
now back to single-spacing,  
which is the default condition.
```

Skipping Lines (^SK)

`^SK [number of lines]`

Default value: *number of lines* = 1

The `^SK` command causes a break in printing. The current line being formatted is printed, the specified number of lines are skipped, then formatting resumes with the next line.

If more lines are specified for skipping than there remain on the page, then the printer simply advances to the top of the next sheet.

`^SK 0` causes the text following the command to be printed on the next line, but no lines are skipped.

Page Numbering (^PN)

`^PN [page number]`

Default value: *page number* = current page number

The formatting routine always keeps track of the total number of printed pages being produced. The `^PN` command causes the page number to be printed on the second line from the bottom of each page. (The page numbers are not printed unless you use `^PN`.)

If `^PN` is used without a parameter, then the current page number, as counted by the formatting routine, is printed. Otherwise, you can specify a starting page number.

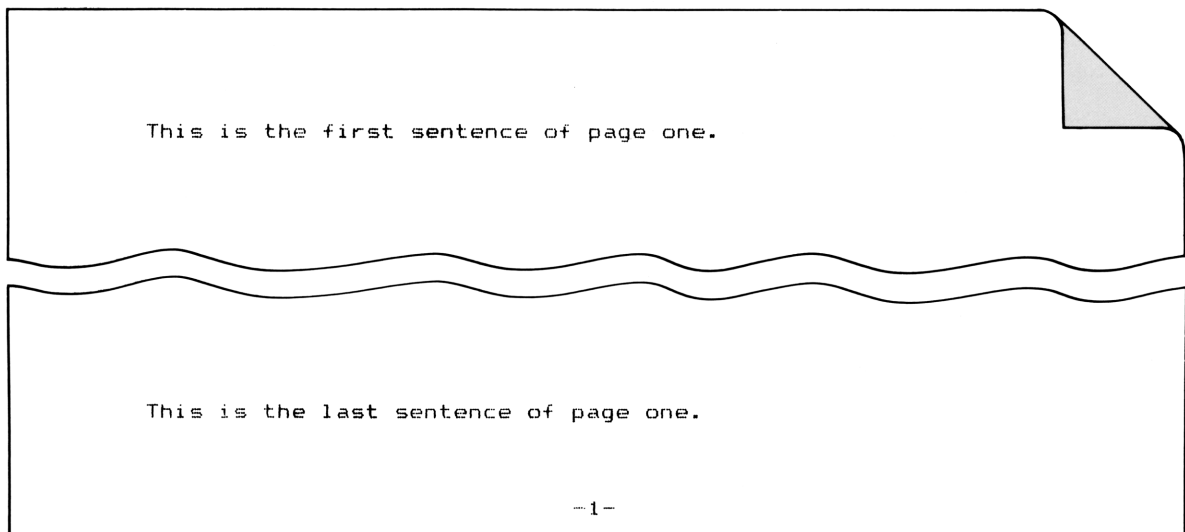
To stop the printing of page numbers, use `^PN 0`. The formatting routine still keeps track of the number of pages being printed, however. If you resume printing page numbers with `^PN` (no parameter), then the page numbers will be the same as if all the printed pages had been numbered.

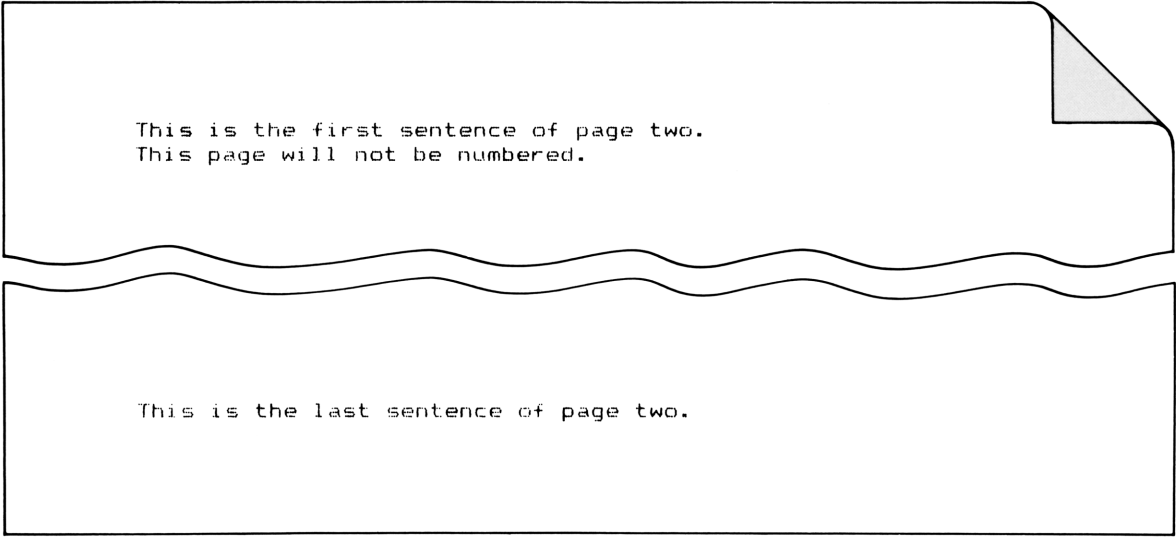
For page numbering to function properly, the page-length setting must be correct (see `^PL`, page 50) and the formatting job must begin with the printer printhead aligned with the first line after a perforation.

Example:

```
^pn ^coThis is the first sentence of page one.
^sk 52This is the last sentence of page one.
This is the first sentence of page two.
This page will not be numbered.
^sk 51 ^pn 0This is the last sentence of page two.
This is the first sentence of page three.
^pn ^sk 52This is the last sentence of page three.
```

The formatted, printed output is:





This is the first sentence of page two.
This page will not be numbered.

The diagram shows a rectangular box representing a page. The top half of the box contains two lines of text. The bottom half of the box is empty. A wavy line separates the top half from the bottom half. The top-right corner of the box is folded over, revealing a grey triangular area.

This is the last sentence of page two.



This is the first sentence of page three.

The diagram shows a rectangular box representing a page. The top half of the box contains one line of text. The bottom half of the box is empty. A wavy line separates the top half from the bottom half. The top-right corner of the box is folded over, revealing a grey triangular area.

This is the last sentence of page three.

Advancing the Page (^AD)

```
^AD
```

The ^AD command causes a break in printing the output, and advances the printer to the first line for printing on the next sheet of paper. For this to work properly, the page length setting must be correct (refer to ^PL on page 50), and the formatting job must start with the printhead aligned with the first line after a sheet perforation. If you have used a negative parameter for the ^PL command (such as PL -60) to set up for a manual, sheet-feed printer, then ^AD will also cause the message `Insert Page...` to appear. (Refer to page 51.)

Using the Escape Command (^ES)

```
^ES
```

The ^ES command is used for advanced formatting techniques that require line lengths longer than the margin limits, control of the line numbers and control of the printer buffer.

When the formatting routine encounters this command, the remaining characters on the line are immediately sent to the printer. Neither the current line count nor the character count are incremented. No end-of-line sequence is sent; the next line of the text will be added to the printer buffer as if the previous line never existed.

This command is not for putting printer escape sequences into your text file. Refer to appendix E for information about how to get printer escape sequences into your text file.

^ES is useful for applications—such as graphics—where it may be necessary to send a line to the printer that is longer than the current margin limits. For other applications, ^ES is useful because the characters following ^ES do not increment the character or line count. When you are sending printer control (escape) sequences, you don't want the character count incremented because these sequences are not part of your printed text. For those cases when you want the line, or character count incremented, refer to the subprograms discussed in appendix F.

Formatting Combinations of Text Files

There are two text-formatting commands that enable you to format several text files at one time. The ^ME (*merge*) command in a text file references another file and merges its output into the first one. The ^DL (*distribution list*) command in a text file specifically references and incorporates entries from a *names file*.

Merging Files (^ME)

If you re-use certain standard pieces of information—like a letterhead or form letter—it is useful to keep the standard information in a single text file, and then reference that file from any other file that needs the standard information. This is called *merging* files. You can merge files any time you want to combine the information from two different text files.

```
^ME filename
```

An ^ME command must be either the only or the last command on a line. No text or commands can follow an ^ME command on the same line.

The ^ME command causes formatting of the edit file to be suspended and formatting of the specified, referenced file to begin. Upon reaching the end of the referenced file, formatting of the original file continues from the text line after the ^ME command.

The merging function can be *nested*. That is, one file can reference a second file for merging, the second file can reference a third file for merging, and so on, up to a fifth file. An error occurs if you try to nest a series of more than five files.

Example: The following three text files will merge during formatting. The merging is nested three files deep.

```
^pa 0 This is FILE1 referencing  
^me file2  
^pa This is the end of the example.
```

FILE1.

```
FILE2. Now FILE2 will reference  
FILE3.  
^sk ^me file3  
Now we will return to FILE1.
```

FILE2.

```
^pa This is a second paragraph.  
It is all part of FILE3.  
Now we will return to FILE2.
```

FILE3.

Be sure that `FILE1` is the edit file when you give the `F` command. `FILE1` will call `FILE2`, which then calls `FILE3`. The printed, formatted output for `FILE1` is:

This is `FILE1` calling `FILE2`. Now `FILE2` will call `FILE3`.

This is a second paragraph. It is all part of `FILE3`. Now we will return to `FILE2`. Now we will return to `FILE1`.

This is the end of the example.

Using a Distribution List (^DL)

A distribution list—familiar to those who write memos—is a list of people who should receive a copy of a certain memo or report. The `^DL` (*distribution list*) command integrates a series of names (a distribution list) from a *names file* into a body of text from an edit file. The `^DL` command can be used, of course, to integrate a list of anything (not just names) into the edit file.

```
^DL filename of names file [, replacement character]
```

Default value: *replacement character* = `` (fK)`.

The `^DL` command creates personalized memos or form letters. When the edit file is formatted, `^DL` produces personalized copies for different people without altering the edit file. Wherever the replacement character (the default ``` or the character specified in the `^DL` command) appears in the edit file, information from an entry in the names file is substituted and as many different copies of the edit file as there are entries in the names file are produced. Within any one copy, the same information is inserted everywhere the replacement character (```) is encountered in the edit file. The `^DL` command must specify the name of the file that has the list of name entries you want to use. You need to specify a replacement character only if you want to use something other than ```.

```
^dl staff
```

References the names file `STAFF` for the distribution list.

A `^DL` command must be either the only or the last command on a line. No text or command can follow a `^DL` command on the same line.

The Names File. To build a names file, simply create a new text file. Use one name entry per text line. The name entry can be as little as a first name or as much as an entire name and address. You can also divide the name entry into *fields* by separating parts of the entry with a replacement character (```). This allows the edit file to call for different parts of a name entry, instead of using the entire name entry for each name insertion. (This is shown in the example below.) You can define up to nine fields in a name entry.

The Replacement Character. The normal (default) replacement character is ``` (the accent grave). To obtain this character, press `[f][K]`. If you want to define a different character for the replacement character, simply specify it along with the `^DL` command.

The same replacement character must be used in the names file that is referenced by a `^DL` and in the edit file that contains the `^DL`.

- In the names file: use ``` to define fields in the name entries. Each entry in a file should have the same kinds and number of fields, but you can define a field as you wish. (In other words, if you define a middle name as the second field in one entry, then all entries should use a middle name in the second field.)

If you don't define any fields, then the entire name entry will be one field. You can define up to nine fields.

```
Becky`Melissa`Hansen
```

This defines three fields: field 1 is the first name, field 2 is the middle name, and field 3 is the last name.

- In the edit file: place ``` in the edit file everywhere that you want an entry from the distribution list to be inserted. If you want to use only certain fields (which must be defined in the names file) of the name entry, then specify a *field number* after the replacement character. For *each field* you want inserted, use *field number*.

```
`1 `3
```

If the current name entry is `Becky`Melissa`Hansen`, then this line in the edit file will be printed as `Becky Hansen`.

The replacement character is recognized as a replacement character *only in the text-filling modes* (Fill and Justify). In the text-copying modes (Copy and Center), the replacement character will be printed as is any other character, with no special meaning.

Example: The following edit file, NOTE, uses the names file, STAFF, to produce a personalized memo from a distribution list.

```
Becky`Hansen
Glen`Mason
Elaine`Peterman
```

File STAFF. The ` defines fields for the first and last names.

```
^dl staff
To: `

^sk 1 From: Judy Martin
^sk 1

`1, there will be a sales
meeting tomorrow at 3:00.
```

File NOTE. References STAFF.

This inserts the entire name entry into the formatted copies.

Every copy will read From: Judy Martin

The ` will insert just the first name field into the formatted copies.

Be sure the edit file is NOTE, not STAFF (the names file), when you give the F command.

The formatted, printed output for NOTE is:

```
To:  Becky Hansen

From:  Judy Martin

Becky, there will be a sales meeting tomorrow at 3:00.
```

To: Glen Mason

From: Judy Martin

Glen, there will be a sales meeting tomorrow at 3:00.

To: Elaine Peterman

From: Judy Martin

Elaine, there will be a sales meeting tomorrow at 3:00.

Example: Multiple fields in a names file can be used to set up the address portion of a standard business letter. Suppose the edit file LETTER uses the names file NAMES.

```
John`Smith`ABC Company`1735 Main
St.`City`State`~`~60637
:
^dl names
^ju ^ta 45 November 16, 1983
^sk 2 `1 `2
^sk 0 `3
^sk 0 `4
^sk 0 `5, `6
^sk 2 Dear `1:
^sk 2 As we discussed in our
last telephone conversation,
:
```

This shows just one entry from the names file NAMES. It has six fields defined in it.*

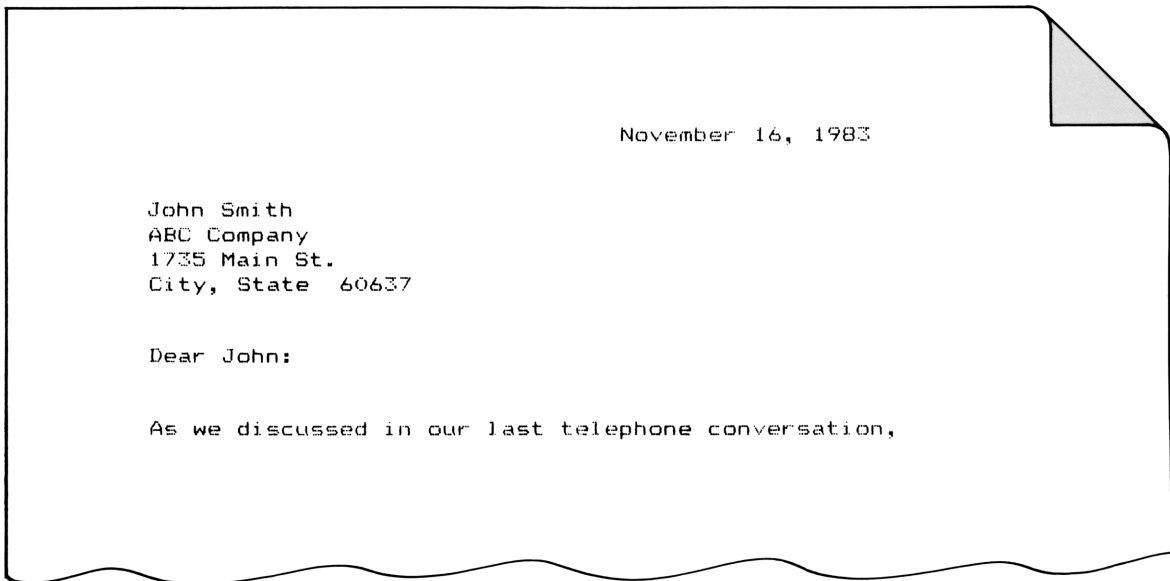
File LETTER, which references NAMES.

Sets Justify mode so that the body of the letter will be justified.

^SK 0 causes the printer to move to the next line.

The body of the letter continues.

The formatted, printed output for LETTER is:



November 16, 1983

John Smith
ABC Company
1735 Main St.
City, State 60637

Dear John:

As we discussed in our last telephone conversation,

* If you put two fixed-space characters between the state and the zip code, the printed version will maintain two spaces before the zip code (refer to page 43).

Owner's Information

Limited One-Year Warranty

What We Will Do

The HP-71 Text Editor Pac is warranted by Hewlett-Packard against defects in materials and workmanship affecting electronic and mechanical performance, but not software content, for one year from the date of original purchase. If you sell your unit or give it as a gift, the warranty is transferred to the new owner and remains in effect for the original one-year period. During the warranty period, we will repair or, at our option, replace at no charge a product that proves to be defective, provided you return the product, shipping prepaid, to a Hewlett-Packard service center.

What Is Not Covered

This warranty does not apply if the product has been damaged by accident or misuse or as the result of service or modification by other than an authorized Hewlett-Packard service center.

No other express warranty is given. The repair or replacement of a product is your exclusive remedy. **ANY OTHER IMPLIED WARRANTY OF MERCHANTABILITY OR FITNESS IS LIMITED TO THE ONE-YEAR DURATION OF THIS WRITTEN WARRANTY.** Some states, provinces, or countries do not allow limitations on how long an implied warranty lasts, so the above limitation may not apply to you. **IN NO EVENT SHALL HEWLETT-PACKARD COMPANY BE LIABLE FOR CONSEQUENTIAL DAMAGES.** Some states, provinces, or countries do not allow the exclusion or limitation of incidental or consequential damages, so the above limitation or exclusion may not apply to you.

This warranty gives you specific legal rights, and you may also have other rights which vary from state to state, province to province, or country to country.

Warranty for Consumer Transactions in the United Kingdom

This warranty shall not apply to consumer transactions and shall not affect the statutory rights of a consumer. In relation to such transactions, the rights and obligations of Seller and Buyer shall be determined by statute.

Obligation to Make Changes

Products are sold on the basis of specifications applicable at the time of manufacture. Hewlett-Packard shall have no obligation to modify or update products once sold.

Warranty Information

If you have any questions concerning this warranty, please contact an authorized Hewlett-Packard dealer or a Hewlett-Packard sales and service office. Should you be unable to contact them, please contact:

- In the United States:

Hewlett-Packard
Personal Computer Group
Customer Communications
11000 Wolfe Road
Cupertino, CA 95014

Toll-Free Number: (800) FOR-HPPC (800 367-4772)

- In Europe:

Hewlett-Packard S.A.
150, route du Nant-d'Avril
P.O. Box CH-1217 Meyrin 2
Geneva
Switzerland
Telephone: (022) 83 81 11

Note: Do *not* send units to this address for repair.

- In other countries:

Hewlett-Packard Intercontinental
3495 Deer Creek Rd.
Palo Alto, California 94304
U.S.A.
Telephone: (415) 857-1501

Note: Do *not* send units to this address for repair.

Service

Hewlett-Packard maintains service centers in most major countries throughout the world. You may have your unit repaired at a Hewlett-Packard service center any time it needs service, whether the unit is under warranty or not. There is a charge for repairs after the one-year warranty period.

Hewlett-Packard products are normally repaired and reshipped within five (5) working days of receipt at any service center. This is an average time and could vary depending upon the time of year and the work load at the service center. The total time you are without your unit will depend largely on the shipping time.

The Hewlett-Packard United States Service Center for battery-powered computational products is located in Corvallis, Oregon:

Hewlett-Packard Company
Service Department

Service centers are maintained at the following locations. For countries not listed, contact the dealer where you purchased your unit.

HEWLETT-PACKARD Ges.m.b.H.
Kleinrechner-Service
Wagramerstrasse-Lieblgasse 1
A-1220 Wien (Vienna)
Telephone: (0222) 23 65 11

HEWLETT-PACKARD FRANCE
Division Informatique Personnelle
S.A.V. Calculateurs de Poche
F-91947 Les Ulis Cedex
Telephone: (6) 907 78 25

HEWLETT-PACKARD NORGE A/S
P.O. Box 34
Oesterndalen 18
N-1345 Oesteraas (Oslo)
Telephone: (2) 17 11 80

HEWLETT-PACKARD BELGIUM SA/NV
Woluwedal 100
B-1200 Brussels
Telephone: (02) 762 32 00

HEWLETT-PACKARD GmbH
Kleinrechner-Service
Vertriebszentrale
Berner Strasse 117
Postfach 560 140
D-6000 Frankfurt 56
Telephone: (611) 50041

HEWLETT-PACKARD ESPANOLA S.A.
Calle Jerez 3
E-Madrid 16
Telephone: (1) 458 2600

HEWLETT-PACKARD A/S
Datavej 52
DK-3460 Birkerød (Copenhagen)
Telephone: (02) 81 66 40

HEWLETT-PACKARD ITALIANA S.P.A.
Casella postale 3645 (Milano)
Via G. Di Vittorio, 9
I-20063 Cernusco Sul Naviglio (Milan)
Telephone: (2) 90 36 91

HEWLETT-PACKARD SVERIGE AB
Skalholtsgratan 9, Kista
Box 19
S-163 93 Spanga (Stockholm)
Telephone: (08) 750 2000

Refer to the address listed under Austria.

HEWLETT-PACKARD (SCHWEIZ) AG
Kleinrechner-Service
Allmend 2
CH-8967 Widen
Telephone: (057) 31 21 11

HEWLETT-PACKARD OY
Revontulentie 7
SF-02100 Espoo 10 (Helsinki)
Telephone: (90) 455 02 11

HEWLETT-PACKARD NEDERLAND B.V.
Van Heuven Goedhartlaan 121
NL-1181 KK Amstelveen (Amsterdam)
P.O. Box 667
Telephone: (020) 472021

HEWLETT-PACKARD Ltd
King Street Lane
GB-Winnersh, Wokingham
Berkshire RG11 5AR
Telephone: (0734) 784 774

International Service Information

Not all Hewlett-Packard service centers offer service for all models of HP products. However, if you bought your product from an authorized Hewlett-Packard dealer, you can be sure that service is available in the country where you bought it.

If you happen to be outside of the country where you bought your unit, you can contact the local Hewlett-Packard service center to see if service is available for it. If service is unavailable, please ship the unit to the address listed above under Obtaining Repair Service in the United States. A list of service centers for other countries can be obtained by writing to that address.

All shipping, reimportation arrangements, and customs costs are your responsibility.

Service Repair Charge

There is a standard repair charge for out-of-warranty repairs. The repair charges include all labor and materials. In the United States, the full charge is subject to the customer's local sales tax. In European countries, the full charge is subject to Value Added Tax (VAT) and similar taxes wherever applicable. All such taxes will appear as separate items on invoiced amounts.

Computer products damaged by accident or misuse are not covered by the fixed repair charges. In these situations, repair charges will be individually determined based on time and materials.

Service Warranty

Any out-of-warranty repairs are warranted against defects in materials and workmanship for a period of 90 days from date of service.

Shipping Instructions

Should your unit require service, return it with the following items:

- A completed Service Card, including a description of the problem.
- A sales receipt or other proof of purchase date if the one-year warranty has not expired.

The product, the Service Card, a brief description of the problem, and (if required) the proof of purchase date should be packaged in adequate protective packaging to prevent in-transit damage. Such damage is not covered by the one-year limited warranty; Hewlett-Packard suggests that you insure the shipment to the service center. The packaged unit should be shipped to the nearest Hewlett-Packard designated collection point or service center. Contact your dealer for assistance. (If you are not in the country where you originally purchased the unit, refer to International Service Information above.)

Whether the unit is under warranty or not, it is your responsibility to pay shipping charges for delivery to the Hewlett-Packard service center.

After warranty repairs are completed, the service center returns the unit with postage prepaid. On out-of-warranty repairs in the United States and some other countries, the unit is returned C.O.D. (covering shipping costs and the service charge).

Further Information

Circuitry and designs are proprietary to Hewlett-Packard, and service manuals are not available to customers. Should other problems or questions arise regarding repairs, please call your nearest Hewlett-Packard service center.

When You Need Help

Hewlett-Packard is committed to providing after-sale support to all of its customers. To this end, our customer support department has established phone numbers that you can call if you have questions about this product.

Product Information. For information about Hewlett-Packard dealers, products, and prices, call:

(800) FOR-HPPC
(800 367-4772)

Technical Assistance. For technical assistance with your product, call the number below:

(503) 754-6666

For either product information or technical assistance, you can also write to:

Hewlett Packard
Personal Computer Group
Customer Communications
11000 Wolfe Road
Cupertino, CA 95014

Error and Status Messages

When you use the Text Editor Module, you may occasionally see messages in the display that are not described in the earlier sections of this manual. These error or warning messages indicate that the program cannot recognize or use an item of input.

Some of the error or warning messages are generated by the HP-71 operating system. These are “system” messages and are explained under “Errors, Warnings and System Messages” in the *HP-71 Reference Manual*.

Other error messages are generated by the Text Editor. This appendix lists and explains these messages. They are presented in alphabetical order. The messages generated by the F (format) command are listed after the messages generated by the other Text Editor commands.

Text Editor Messages

Done: *filename*

- The Text Editor has been exited and the HP-71 is now ready for the next task.

File Exists: *filename*

- The filename to which you want your lines to go already exists. Use the + option, or a new filename.

File Protect

- Your `keys` file is secured. The Text Editor is unable to run because it can't copy your `keys` file for temporary storage in the file `EDUKEYS` (refer to appendix D). If you want your `keys` to be merged with the Text Editor keys, unsecure your `keys` file. If not, you can rename the file or copy it out of main memory then purge it.
- The operation you are attempting (D (delete), I (insert), R (replace) or T (text)) is not valid because your edit file is secured. Unsecure the edit file.

Insufficient Memory

- The Text Editor program checks the amount of available memory for the files that are being manipulated. If the amount of available memory is low, this message appears then the Command Prompt returns to the display. Check your file to determine whether or not the command was completed. If not, free up some memory by purging some files as described in section 6 of your *HP-71 Owner's Manual*. If there is insufficient memory for *any* operation, the Text Editor exits. The minimum byte requirement to run `EDTEXT` is 2300 bytes.

Invalid File Type: *filename*

- The filename in the command string must be a text file.

Invalid Parm: *parameters*

- The Text Editor does not recognize the parameter portion of the command string. Review the syntax for your command.

If you are using M (move) or C (copy), the block you are trying to move or copy contains the current line and it is not the first or last line of the block. Change the location of the current line to outside the block, or to the first or last line of the block.

Line Too Long

- In text mode, a line longer than 96 characters cannot be edited.

Not Found

- The search string in an S (search) or R (replace) command was not found in the file.

OK to Delete? Y/N:

- The line parameters used in the D (delete) command indicate more than one line to be deleted. The Text Editor checks to be sure you wish to do this. You must respond with ☐ Y for the deletion to occur, or with ☐ N to return to the Command Prompt.

Working...

- R (replace), M (move), C (copy) and D (delete) generate this message when the command takes time to complete.

? Cmd: *command*

- The program does not recognize the letter as a valid Text Editor command. The valid commands are C, D, E, F, H, I, L, M, P, R, S and T.

Text Formatting Messages

The F (format) command produces certain messages under specific conditions. Some of these are merely status messages, while others occur in response to an error. An incorrectly typed or constructed formatting command will produce an error message. In most cases, an error message does not halt printing.

Status and error messages can appear either in the HP-71 display or on the printed output.

Displayed Messages

These messages, listed in alphabetical order, appear in the display:

File *filename* Not found

- The specified file is not in memory. Check the catalog to see what filenames exist.

Insert Page ...

- If a sheet-feed printer is in use, change the paper and press **END LINE** to continue.

Insufficient Memory

- There is not enough available memory to execute the F (format) command. You may need either to purge some files to make room (refer to section 6 in the *HP-71 Owner's Manual*) or exit the Text Editor and call the formatting routine (EDFORM) from outside the Text Editor (refer to appendix F). The minimum byte requirement to run EDFORM is 2500 bytes.

String Ovfl

- The formatting routine attempted to read a line longer than 250 characters. Use the BASIC Keywords in appendix C to modify the long line.

Working...

- The F (format) command is executing.

Printed Messages

The following messages, in alphabetical order, appear on the printer:

Invalid Parm: ^*command* *first parameter* [*second parameter*]

- A valid command name has been used with an invalid parameter, such as setting the left margin in a higher-numbered column than the right margin. The command and parameters you supplied are printed.

Merge > 5 Files: ^ME *filename*

- The nesting limit of merge files has been reached. Up to five levels of merging are permitted.

Mult Dist Lists: ^DL *filename*

- Only one distribution list file may be used in any particular document.

? Cmd: ^*command*

- A command has been mistyped. Formatting will continue without altering any of the modes.

Text Editor Module BASIC Language Enhancements

There are nine keywords in the Text Editor Module that you can use from the keyboard or in your own BASIC language programs.

To use these keywords (except `EDTEXT`), you must have either the Text Editor Module installed in your HP-71 or you must copy the `EDLEX` file which contains the keyword definitions into the HP-71 memory using the `COPY` command (refer to section 6 of the *HP-71 Owner's Manual*). If you attempt to use one of the keywords without the keyword being available from the module or in the HP-71 memory, an error message will be displayed. The Text Editor module must be in place when you use `EDTEXT` because the program calls subprograms in the module.

The following is a list of the keywords, their syntax and an example of the use of each. In most cases, the parameters are numeric expressions, string expressions or literals (refer to section 3 of the *HP-71 Owner's Manual*).

```
DELETE# channel number,record number
```

The `DELETE#` statement deletes the specified record from the text file associated with the *channel number*. Channel numbers are assigned with the `ASSIGN#` statement (refer to section 14 of the *HP-71 Owner's Manual*). Record numbers always begin at 0, so line number 1 is record number 0.

The *channel number* and the *record number* are numeric expressions, rounded to integer values.

`DELETE#` generates an error message if the assigned file is external, protected or not a text file.

Example: `DELETE# 11,14` deletes record number 14 from the file associated with channel 11.

```
EDTEXT filename [,command string]
```

The `EDTEXT` statement starts the Text Editor program. The *filename* parameter specifies the file to be edited. If you use the optional *command string* parameter, the Text Editor begins immediate execution of the Text Editor commands in the *command string*. The *command string* may be a string expression or literal.

When running EDTEXT from a BASIC program, you should have your program check to see if any errors have occurred while in EDTEXT. You can have your BASIC program read the display using the DISP# function, and compare the string to Done. If there is a match, you know the Text Editor successfully completed running. If there is no match, you know there was a problem when you ran the Text Editor.

Examples: EDTEXT FROGS begins running the Text Editor program with the file FROGS as the edit file.

EDTEXT FROGS,L begins running the Text Editor program with the file FROGS being listed to the display device.

If your HP-71 is version 1AAAA or 1BBBB (to find out the version, execute VER#—refer to appendix A, *HP-71 Owner's Manual*), always place an END or STOP statement at the end of the program containing EDTEXT, or your memory may be altered. Also, if you suspend a program, invoke EDTEXT creating a new file, then resume the original program, the original program must end with END or STOP.

FILESZR (*filename*)

The FILESZR function returns the number of records in the specified file if that file exists. The *filename* parameter is a string expression. If an error is detected (filename does not exist, for example), the negated error number is returned so that you can tell the difference between an error and the number of records. If *filename* contains an illegal port specifier, such as FROGS:PORT(8), the error message Invalid Filespec is generated.

Example: FILESZR ('FROGS') returns the number of records in the file FROGS.

INSERT# *channel number,record number;new record*

The INSERT# statement inserts the *new record* immediately before the specified *record number* in the file associated with the specified *channel number*. The *channel number* must first be assigned to the file using the ASSIGN# statement (refer to section 14 of the *HP-71 Owner's Manual*). Record numbers always begin at 0, so line 1 is record 0.

The *new record* must be a string expression. The *channel number* and the *record number* are numeric expressions, rounded to integer values.

INSERT# generates an error if the file is external, protected or not a text file.

Example: `INSERT# 11,35;"This is the new line being inserted."` inserts the string before record 35 (line 36) of the file associated with channel 11. The old record 35 becomes record 36.

`LIST filename [begin line [end line]]`

The `LIST` statement lists a text file. Depending on the parameters you specify, it lists either the entire file, a single line, or a range of lines. Line numbers are specified using integer constants. The line number parameters are optional, and the whole file is listed if they are not included. Refer to `LIST` in the *HP-71 Reference Manual* for details.

`MSG# (number)`

The `MSG#` function returns the corresponding system message. The *number* parameter specifies which system message to display. Refer to “Errors, Warnings, and System Messages” in the *HP-71 Reference Manual*.

Example: `MSG# (72)` will display `Too Slow`.

`REPLACE# channel number,record number;new record`

The `REPLACE#` statement replaces the record indicated by *record number* with the *new record*. The *channel number* must first be assigned to the file by using the `ASSIGN#` statement (refer to section 14 of the *HP-71 Owner's Manual*). Record numbers always begin at 0, so line number 1 is record 0.

The *new record* is a string expression. The *channel number* and the *record number* are numeric expressions rounded to integer values.

`REPLACE#` returns an error if the file is external, protected or not a text file.

Example: `REPLACE# 11,35;"This line will replace the old line."` replaces record 35 (line 36) of the file associated with channel 11. Old record 35 no longer exists.

```
SEARCH (search string,column,begin line,end line,channel number)
```

The `SEARCH` function searches the file associated with the indicated *channel number* for the *search string*, beginning with the specified *column* and *line number*. The search continues through the *end line* specified. If the search is successful, `SEARCH` returns a value in the form `nnn.ccclll`, where `nnn` is the record number, `ccc` is the column number and `lll` is the length of the matched string. If the search is unsuccessful, `SEARCH` returns a zero.

The *search string* can be any string expression, and can contain the special pattern characters discussed in section 2 beginning on page 33. The other parameters are numeric expressions rounded to integer values.

Example: Suppose that channel 11 has been assigned to the file `FROGS` and the string `frogs are green` appears beginning in column 8 of line 36.

```
A=SEARCH ("frogs are green",1,1,9999,11)
```

searches the file `FROGS`, beginning with column 1 of line 1 through line 9999, for the search string and returns the value 35.008015 in `A`. (Remember that record numbers start with 0, so line 36 is record 35).

```
SCROLL column
```

The `SCROLL` statement repositions the message in the display. The indicated *column* of the message is now in position one of the display. *Column* can be specified by any numeric expression and is rounded to an integer value. The maximum value for *column* is 96 (the buffer size) minus the window size.

Text Editor Filenames

The Text Editor module contains several files, and each file has a name. These names (except EDLEX) must not be used as the names of files in the HP-71 user memory, as the HP-71 first searches its own memory for a file before searching the plug-in modules. EDLEX is never called by the Text Editor. The following list gives the name of each file in the module, along with a brief description of the file.

EDTEXT	The Text Editor BASIC language program.
EDLEX	A LEX file containing the assembly level support for the Text Editor, including the BASIC keywords.
EDKEYS	The Text Editor keys file.
EDROPR	A BASIC file containing the text formatting reading and printing subroutines.
EDUKEYS	A temporary keys file created by the Text Editor in main memory to store your user defined keys while the Text Editor is running. When you exit the Text Editor, these keys again become your current keys file.

Using Printer Escape Sequences

Hewlett-Packard printers share a common standard for *escape sequences*. Escape sequences are special character sequences used for controlling printer operations. These sequences must be included in your text file, just like text formatting commands, to control the printer, but you can't enter these sequences in text mode. You can't type an escape sequence into the display because the display will act on it.

Escape sequences let you set alternate printing modes (such as expanded or compressed print), vertical line spacing, page length, and so on. With a little practice, you can learn to enhance the appearance of your documents by using the features that your printer provides. For instance, you can emphasize a heading at the top of a page by printing in expanded mode, which prints characters at twice their normal width.

It is not possible to enter control or escape sequences into a text file while in the Text Editor. One way to get these sequences into your file is to assign a channel number to a text file, write the escape sequence to the file using `PRINT#`, close the file then use the merge command (`^ME`) in the Text Editor when the escape sequence file is needed.

Example: The escape sequence for the HP-82905B printer double-density graphics mode is `ESC &k2S`. To use this feature, you could:

1. Create a file called `DOUBLED: CREATE TEXT DOUBLED.`
2. Assign a channel number: `ASSIGN#6 TO DOUBLED`
3. Print the escape sequence to the file: `PRINT#6; CHR$(27)&"&k2S"`
4. Close the file: `ASSIGN#6 TO *`

`DOUBLED` now exists and you can use `^ME DOUBLED` to cause the lines following the command to be printed in double-density print. `^ME` assumes you are dealing with entire words, so if you try to emphasize a part of a word, you will get spaces in the middle of the word.

A second way to put escape sequences into your file is to use the `R` (replace) command. When you input your text file, put a dummy character every place you need an escape sequence. Then replace this character.

Example: Suppose you used the character `<` in your file every time you wanted double density mode. To replace `<` with the proper escape sequence, do the following:

1. Create a string equivalent to the replace command you want to execute:

```
A$ = ".#R/(/"&CHR$(27)&"&k2S"
```

2. Call the Text Editor, using the filename and `A$` as the parameters: `EDTEXT filename, A$`

Once the replace command is completed, you shouldn't edit any line containing the escape sequence. If you look at the line, the escape sequence will not show, and the cursor may not appear in the correct position. If you need to edit the file, reverse the above procedure, replacing `CHR$(27)&k2S` with the dummy character.

This method could also be used to insert only the escape character `CHR$(27)`. Any place that you want an escape sequence, type in your dummy character (`(`) followed by the last part of the command (`&k2S` for example). After you finish editing the file and are ready to print it, make the replacement described above, with the statement: `A$ = ".#R/(/"&CHR$(27)`.

Text Formatting Subprograms

EDFORM is the text formatting routine. EDITFPR1, EDITFPR2, EDITFPR3 and EDITFRD are subprograms used by the text formatting program to control reading and printing.

- SUB EDFORM (*filename, # of copies, fixed-space character*)

EDFORM is the text formatting routine that the Text Editor calls when you use the command F. This program can also be run from outside the Text Editor, called either from the keyboard or from a BASIC program. If you get an `Insufficient Memory` error message when trying to execute the F command from within the Text Editor, exit the Text Editor and call the formatting routine directly.

The *filename* and *fixed-space character* parameters are string expressions. *# of copies* is a numeric expression, rounded to an integer value. All parameters must be included. You can use the null string (") for the fixed-space character.

Example: Call the text formatting program to format the file `SAMPLE`, print 2 copies, and use the default fixed-space character.

```
CALL EDFORM('SAMPLE',2,'')
```

If your HP-71 is version 1AAAA or 1BBBB, always place an `END` or `STOP` statement at the end of the program containing the call to EDFORM, or your memory may be altered.

You can change what the text formatting program does by changing the four read and print subprograms. If you have a subprogram by the same name in main memory, your subprogram will be executed instead of the subprogram in the Text Editor. In this way, you can control how information is printed and read. To view these four subprograms, list or print `EDRDPFR`, a BASIC file in the Text Editor module.

The four subprograms, their parameters and descriptions are:

- SUB EDITFPR1(*string of blanks defining left margin, formatted text*)

This is the main print routine. You could write your own routine to write to something other than a printer, or to do something special with the output.

- SUB EDITFPR2(*left margin, right margin, page number*)

This routine controls the printing of the page number at the bottom of the page. The three parameters are numeric expressions rounded to integer values. The text formatting routine sends this information to EDITFPR2 so if you change one of the values, the new value is returned to the formatting routine.

- SUB EDITFPR3(*string of characters including ^ES*,*line number*)

This routine handles the use of the ^ES command. The first parameter is a string expression and the second is a numeric expression. EDITFPR3 does not increment the line number.

- SUB EDITFRD(*designated channel number*,*l\$*,*l1*)

EDFORM uses this routine to read from text files. It sequentially reads the file associated with the *designated channel number*, and the corresponding string is returned in *l\$*. The *designated channel number* is mapped into local channel #1. *l1* acts as a flag, indicating if any error occurred—if set on exit, some error (such as End of File) occurred.

Command Summary and Index

Text Editor Commands

[b[e]] C [filename]

(page 24)

Default values: current file *b* = current line
e = *b*

other file *b* = line 1 of other file
e = last line of other file

Copy. Copies a specified range of lines inserting them before the current line.

[b[e]] D [filename[+]]

(page 27)

Default values: *b* = current line
e = *b*

Delete. Deletes a specified range of lines.

E

(page 22)

Exit. Exits the editing session.

F [copies][fixed-space character]

(page 35)

Default values: *copies* = 1
fixed-space character = ~ f =

Format. Invokes the text formatting routine.

H

(page 36)

Help. Displays the Text Editor commands syntax.

[*line number*] I

(page 25)

Default value: *line number* = current line

Insert. Puts the Text Editor into insert mode so that the user can insert lines of text between existing lines.

[*b*][*e*] L [*n*][H]

(page 22)

Default values: *b* = current line
e = last line

List. Lists lines of the edit file to the current display device.

[*b*][*e*] M [*filename*]

(page 24)

Default values: Current file *b* = current line
e = *b*

Other file *b* = line 1 of other file
e = last line of other file

Move. Moves a specified range of lines, inserting them before the current line and deleting the original lines.

[*b*][*e*] P [*n*][H]

(page 22)

Default values: *b* = current line
e = last line

Print. Prints lines of the edit file to the current printer device.

[*b*][*e*][?]R/*string1*/*string2*[/]

(page 28)

Default values: *b* = current line
e = *b*

Replace. Replaces a string or strings in the edit file with a specified string.

`[b[e]][?]` `S/string[✓]`

(page 28)

Default values: *b* = current line plus 1
e = last line

Search. Search for a specified string in the edit file.

`[line number]` `T`

(page 19)

Default value: *line number* = current line

Text. Puts the Text Editor into text mode by placing the current line into the display for character editing.

Special Characters as Line Number Parameters

For any parameter that is a line number, there are two characters that can be used as substitutes for a line number.

- The period `.` means the current line of the file.
- The pound sign `#` means the last line of the file.

Concatenation of Commands

The Text Editor commands can be concatenated (placed in sequence) by using the semi-colon `;` as the delimiter between commands. If one of the commands contains an error, the commands prior to the error will be executed, then the remaining commands will return to the display to be corrected.

Special Pattern Characters

The backward slash `\` (`f` `l` when in the Text Editor) is used as the switch to turn on and turn off the special pattern feature in search and replace strings.

Five characters have special meaning to the program when this feature is turned on.

Period character `.`

(page 33)

- Signifies the search for a “wild card” character.

Commercial “at” character `@`

(page 33)

- Signifies the search for an unspecified number of “wild card” characters.

Ampersand character `&` [\(page 33\)](#)

- Every place that the ampersand appears in *string2*, it is replaced by the string matched by *string1*. *String2* is then inserted into your file.

Up-arrow character `^` [\(page 34\)](#)

- Signifies the search for a string only at the beginning of a line of text.

Dollar sign character `$` [\(page 34\)](#)

- Signifies the search for a string only at the end of a line of text.

Text Formatting Commands

This index is an alphabetical listing of all fifteen HP-71 text formatting commands. Each entry gives a short description of the command: its syntax, its default parameter values, and a page reference to the main text.

`^AD` [\(page 56\)](#)

Advancing the Page. Causes a break in printing the output and advances the printer to the first line for printing on the next sheet of paper.

`^CE` [*page-test parameter*] [\(page 44\)](#)

Center Mode. Copies (as in Copy mode) and centers the subsequent text from the text file, line by line. Leading and trailing spaces on a line are ignored during formatting. Center mode lasts until a `^FI`, `^JU`, `^CO`, `^PA`, or `^TA` command is executed. `^PA` or `^TA` will set Fill mode.

`^CO` [*page-test parameter*] [\(page 44\)](#)

Copy Mode. Causes the subsequent text from the text file to be copied line by line exactly as it appears on the text-file line. Trailing spaces are ignored, but leading spaces are preserved. Copy mode lasts until an `^FI`, `^JU`, `^CE`, `^PA`, or `^TA` command. `^PA` or `^TA` will set Fill mode.

[^]DL *filename of names file* [*, replacement character*]

(page 58)

Default value: *replacement character* = ` (f k)

Distribution List. Integrates a series of entries from a names file into the edit file. Wherever a body of text formatted from the edit file uses the replacement character, information from the names file is substituted. The formatting routine produces as many different formatted versions of the edit file as there are entries in the names file. Within any one copy, the same information is inserted everywhere the replacement character is encountered.

No text or command can follow a [^]DL command on the same line. You can use only one [^]DL per text file.

In the names file, you can specify fields within each name entry by separating parts with the replacement character.

```
Becky` Melissa` Hansen
Dear Ms. `3:
```

This name entry in the names file has three fields. This command in the edit file will be formatted as Dear Ms. Hansen:

[^]ES

(page 56)

Escape Mode. Escape mode is used for advanced formatting techniques that require line lengths longer than the margin limits, control the the line numbers, and control of the print buffer.

[^]FI [*page-test parameter*]

(page 41)

Fill Mode. The default text mode. Fill mode produces full lines of filled, formatted text. Any extra spaces in your text file—leading, trailing, or intermediate—are ignored.

[^]JJ [*page-test parameter*]

(page 41)

Justify Mode. Causes the formatted text to be filled (as in Fill mode) *and* right-justified.

[^]MA [*left margin* [*right margin*]]

(page 49)

Default values: *left margin* = 13
right margin = 72

Margins. Sets left and right margins to the numbered columns.

^ME *filename***(page 57)**

Merge. Causes the formatting program to suspend formatting of the edit file and begin formatting the specified file. You can nest up to five files for merging. No text or commands can follow a ^ME command on the same line.

^PA [*number of spaces to indent*]**(page 46)**

Default value: *number of spaces* = 5 or the last value used.

Paragraph. Starts a new line of printed text, indenting it the given number of spaces. If no number is given, it indents the same number as it did the last time ^PA was used. After indenting, ^PA restores Fill or Justify mode, whichever was last in effect. If Copy or Center mode was last in effect, ^PA cancels it and sets Fill mode.

A negative indentation parameter moves the beginning of the paragraph to the left of the prescribed margin. The command ^PA 0 does not indent but skips a line instead.

^PA includes an automatic page test for two available lines on the sheet.

^PL [*page length* [*top margin* [*bottom margin*]]]**(page 50)**

Default values: *page length* = 66
top margin = 6
bottom margin = 6

Page Length. Tells the formatting routine how many lines of print are possible per page (the total page length), and how many lines to leave as margins on the top and bottom of the page. For a sheet-feed printer, use a *negative* number for the page length parameter.

^PN [*page number*]**(page 53)**

Default value: *page number* = current page number

Page Numbering. The text formatting routine always keeps track of the page numbers, but they are not printed until the command ^PN occurs. ^PN 0 halts the printing of page numbers.

^SK *[number of lines]*

(page 53)

Default value: *number of lines* = 1

Skip Lines. Causes a break in printing on the current line of output, then skips the given number of lines on the formatted page. **^SK 0** causes a break without skipping any lines.

^SP *[number of lines]*

(page 52)

Default value: *number of lines* = 1

Spacing. Sets the spacing between lines of printed text output. The number of lines of spacing can be from 1 to 5. This command does not cause a break in printing.

^TA *column number*

(page 47)

Tab to a Column. Indents the text being printed to the given column position on the same line, if possible. After the tab, Fill or Justify mode—whichever one was last in effect—is restored. However, the current printed line (the one being tabbed) is never justified. If Copy or Center mode was last in effect, **^TA** cancels it and sets Fill mode.

Subject Index

Page numbers in **bold type** indicate primary reference; page numbers in regular type indicate secondary references.

A-B

Accent grave (`), 58, **59**, 60
Advancing paper in printer, 51, 56
Ampersand (&), **33**
Appending text, 19, 24-25, 27-28
ATTN, 13
Blanks, trailing and leading, **41**, 44
Breaking (printing), **44**, 52, 56

C

C command. *See* Copy command
Cautions, **9**
Center mode, 41, **44-46**
Circumflex (^), 34, 39
CMD :, 12
Colon, **42**
Command
 breaking, **44**
 character, **39**
 concatenating, 17, **36**, 39
 embedded, 3, 7, 35
 prompt, 10, 12, 14
 syntax
 Editor, **18**
 Formatter, **39**
Commercial “at” symbol (@), **33**
Conventions, in owner’s manual, 18, 41
Copy
 command, 17, **24-25**, 70
 default values, 24
 mode, 41, **44-45**
 duration of, **45**
Copying, 41, **44**
Correcting errors, 12, 20
Creating a file, **11**
Current line, 11, 13, 19, 25, 28

D

D command. *See* Delete command
Default value, 17, 40
DELETE #, **73**
Delete command, 17, **27-28**, 69, 70
 default values, 27
Delimiters, **28**, 31, 36
Display window setting, **22**
Distribution list, 57, **58-62**
Dollar sign (\$), **34**
DOUBLED, **79**

E

E command. *See* Exit command
EDFORM, 71, **81**
Edit file, 57
EDITFPR1, **81**
EDITFPR2, **81**
EDITFPR3, **82**
EDITFRD, **82**
EDKEYS, **77**
EDLEX, 73, **77**
EDROPR, **77**, 81
EDTEXT, **11**, 19, 23, 25, 42, 69, 73-74, 77
EDUKEYS, 10, 69, 77
End of page, testing for, **46**, 51
END LINE, 11, 12, 14, 16, 19, 21
Entering text, **12**
Eof, 11, 12, 13
Error messages, 7, **69-71**
Escape commands, **56**
Escape sequences, 56, **79-80**
Exclamation point, **42**
Exit command, 16, 17, **22**, 42

F

F command. *See* Format command
Fields, **59**

FILESIZE, **74**
 FILE1, FILE2, FILE3, **57-58**
 Filename, **12, 22, 40, 70**
 Fill mode, **41-44**
 Filling, **41**
 Fixed-space character, **35, 43, 81**
 Format command, **17, 26, 35, 40, 42, 81**
 Formatting modes, **41**
 Form letters, **58**

H

H command. *See* Help command
 Help command, **17, 36-37**
 Hyphenation, **41**

I

I command. *See* Insert command
 Improper exit, recovering from, **22**
 Indenting
 by tabs, **47-48**
 lines, **46**
 paragraphs, **46-47**
 Information
 product, **67**
 technical assistance, **67**
 INSERT #, **74-75**
 Insert
 annunciator, **26**
 command, **15, 17, 25-27, 69**
 default values, **25**
 lines into a file, **15**
 Installing the module, **9**

J-K

Justification, **41, 44**
 Justify mode, **41-44**
 Key redefinitions, **10**
 Keyboard overlay, **10**
 keys file, **69**

L

L command. *See* List command
 Left arrow annunciator, **14**
 LETTER, **62**
 Letters, upper- and lower case, **12, 18, 29, 39**
 Line
 length, **45**
 numbers, **12, 13, 14, 18, 23**
 substitute characters for, **18**
 scrolling through a, **14, 15**

Lines

 format of, **46**
 skipping, **47, 50, 51, 52, 53**
 spacing, **52-53**
 LIST, **75**
 List Command, **17, 22-23, 25**
 Default values, **23**

M

M command. *See* Move command
 Margins
 left and right, **49-50**
 top and bottom, **50**
 MEMO, **25**
 Memory requirements, **69, 71**
 Memos, **58**
 Merging files, **57-58, 71, 79**
 Messages, error and status, **69-72**
 Move command, **17, 24-25, 70**
 Default values, **24**
 Moving around in a file, **14**
 Mnemonics, **39**
 MSG#, **75**

N-O

[N], **27, 29-32, 70**
 NAMES, **62**
 Names file, **57, 58, 59**
 Nested files, **57, 71**
 NOTE, **60-61**
 Numbering of pages, **54**
 Owner's Manual
 HP-71, **7, 14, 22, 69, 71, 73, 74, 75**
 HP 82401 HP-IL Interface, **7**

P

P command. *See* Print command
 Page
 end of, **46**
 format of, **48**
 length, **46, 50-52**
 negative parameter, **51**
 numbering, **50, 53-55**
 width, **48-50**
 Page-text parameter, **41, 42, 47**
 Paper, roll or fanfold, **51**
 Paragraph, new, **46-47**

Parameters, 17-18, 40, 41
 characters to be used in place of, 18
 editor command, 17-18
 formatting command, 40
 numeric, 18, 40
 optional, 18, 40
 separating, 18, 40

Perforation
 page, 46
 skipping, 51

Period, 18, 33, 42

Pound sign (#), 18

Print command, 17, **22-23**
 Default values, 23

Printer
 Escape sequences, 56
 roll-fed, 52
 sheet-feed, 51, 56
 thermal, 51

Purging a file, 22

Q-R

[Q], **29-32**

Question mark, 42

Question mark option, **29-32**

R command. *See* Replace command

Reference manual, for the HP-71, 69, 75

Removing the module, 9

Referencing files for merging, 35

Repair, 65-67

REPLACE\$, 75

Replace command, 17, **28-32**, 69, 70, 79-80
 Default values, 28

Replacement character, 58, **59**

[RUN], 13

S

S command. *See* Search command

SAMPLE, **11-16**, 19, 23, 25, 35

SCROLL, 76

Scrolling a line, 14, 15
 speed, 14

SEARCH, 76

Search command, 17, **28-32**, 70
 default values, 28

Semi-colon, 36

Service, **64-67**

Skipped lines. *See* Lines, skipping

Space, 41, 43, 44
 extra, 42

fixed, 35, 43

Special pattern characters, **33-34**

STAFF, **60-61**

Status messages, **69-71**

Syntax, 18, 36, 41

T

T command. *See* Text command

Tab, **47-48**

TEST, 42

Text

command, 12, 17, **19-22**, 25, 42, 69, 70
 default values, 19

copying. *See* Copying

file, 42, 70

files, combining, **57-58**

filling. *See* Filling

modes, 41

default, 42

Tilde (~), 35, 43

Typing aids, 10

V-Y

Version, HP-71, 74, 81

Warranty, **63-64**

Wildcards, 33

Words, keeping together, 43

[Y], 27, 29-32, 70

How To Use This Manual (page 7)

- 1: Getting Started (page 9)**
- 2: Text Editor Commands (page 17)**
- 3: Text Formatting Commands (page 39)**
- A: Owner's Information (page 63)**
- B: Error and Status Messages (page 69)**
- C: Text Editor Module BASIC Language Enhancements (page 73)**
- D: Text Editor Filenames (page 77)**
- E: Using Printer Escape Sequences (page 79)**
- F: Text Formatting Subprograms (page 81)**
- Command Summary and Index (page 83)**



**Portable Computer Division
1000 N.E. Circle Blvd., Corvallis, OR 97330, U.S.A.**

**European Headquarters
150, Route du Nant-D'Avril
P.O. Box, CH-1217 Meyrin 2
Geneva-Switzerland**

**HP-United Kingdom
(Pinewood)
GB-Nine Mile Ride, Wokingham
Berkshire RG11 3LL**