HP TechWriter's User's Guide
for the HP 9000 Series 200 Computers

Manual Part No. 98819-12111

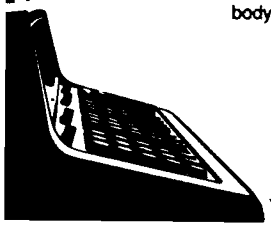
Disc Part No.

Disc Boot:
98819-12314 3½" external
98819-12514 5¼" internal
98819-12614 5¼" external

Disc File:
98819-12315 3½" external
98819-12515 5¼" internal
98819-12615 5¼" external

Disc EDIT:
98819-12316 3½" external
98819-12516 5¼" internal
98819-12616 5¼" external

Disc LIST:
98819-12317 3½" external
98819-12517 5¼" internal
98819-12617 5¼" external



Hewlett-Packard Company
3404 East Harmony Road, Fort Collins, Colorado 80525

Printing History

New editions of this manual will incorporate all material updated since the previous edition. Update packages may be issued between editions and contain replacement and additional pages to be merged into the manual by the user. Each updated page will be indicated by a revision date at the bottom of the page. A vertical bar in the margin indicates the changes on each page. Note that pages which are rearranged due to changes on a previous page are not considered revised.

The manual printing date and part number indicate its current edition. The printing date changes when a new edition is printed. (Minor corrections and updates which are incorporated at reprint do not cause the date to change.) The manual part number changes when extensive technical changes are incorporated.

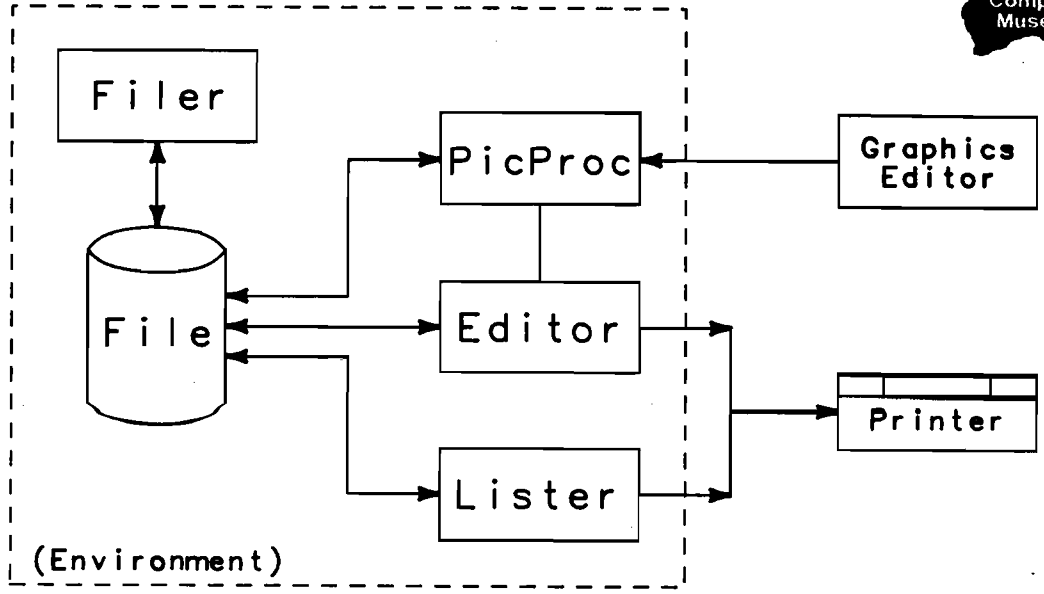
June 1984...First Edition

Table of Contents

Software Installation

Chapter 1

HP TechWriter is a package of programs which aids you in the creation of documents. The logical structure of these programs is shown below:



- The Editor allows you to create documents.
- The Picture Processor translates plot files into a form the Editor and Lister can read. The plot files may be created with any graphics editor that works with HP TechWriter. Call your local Hewlett-Packard Sales Representative for a list of supported graphics editors.
- The Lister takes up to fifty files and lists them as one document. It can also make a table of contents from commands in the files.
- The Filer is a program which allows you to move, rename, copy, and purge files, and more. Also, whole directories can be created, named, copied, and deleted.

Your HP TechWriter software package came with the following parts:

- Boot disc `BOOT:`. This contains the environment to run the HP TechWriter package. If you have a Pascal operating system, you can use the `BOOT:` disc instead of your Pascal discs if you want only the HP TechWriter programs.
- Software disc `EDIT:`. This contains the HP TechWriter Text Editor, a configuration file, a stream file which installs a codeword into the configuration file, and a file which allows the Engineering Graphics System (EGS/200) Manager to access the HP TechWriter programs. (If you don't have EGS/200 you don't need this file.)
- Software disc `LIST:`. The Lister and the Picture Processor are on this disc. It also contains several example picture files: one is an unprocessed picture file, and three are processed picture files. We use the unprocessed picture file in our example of how to use the Picture Processor. In addition, there is a text file called `EXAMPLE.TEXT` which loads the processed picture files in so you can see what pictures look and act like.
- Software disc `ACCESS:`. The Filer and the Mediainit programs are on this disc. Both of these programs deal with file- and directory-manipulation functions.
- Software manual *Pascal 3.0 User's Guide.* This describes basic aspects of the Pascal environment, under which HP TechWriter operates. This is an important reference document. It includes information on handling discs and solving problems that may occur. *If you own a Pascal 3.0 operating system that you would like to use instead of the HP TechWriter Environment, please read the first four chapters of the Pascal 3.0 User's Guide before trying to use this manual.*
- Software manual *HP TechWriter User's Guide.* This contains information on installing and using your HP TechWriter system. The manual contains a tutorial in which you actually make a document, various techniques for using the package, and a reference section with details on all commands and features. This is what you're reading now.

If You Are Running HP TechWriter By Itself...

HP TechWriter provides an environment from which you can run the HP TechWriter programs. The process of loading an environment into a computer is called "booting". This section contains instructions for booting your HP TechWriter Environment.

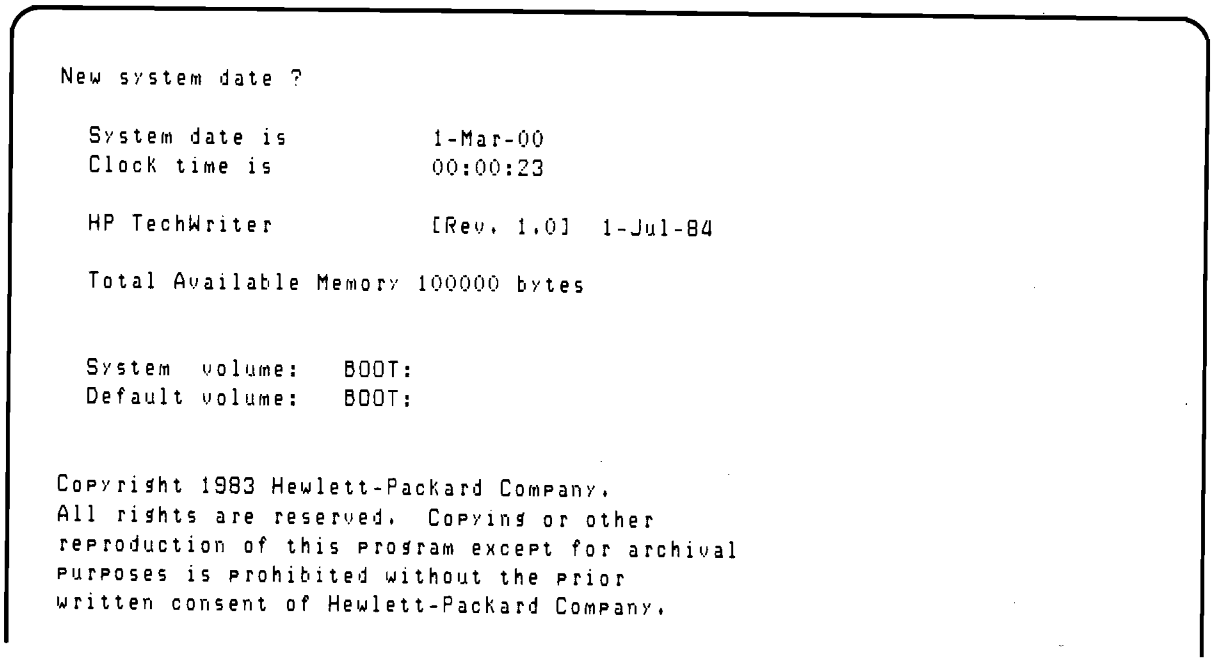
Note

If you do not know how to use flexible discs, please read the sections called *Flexible Disc Handling Guidelines* and *Inserting a Disc Into Your Disc Drive* in the *Pascal 3.0 User's Guide* before proceeding any further.

Booting HP TechWriter

1. If you are using an external disc drive, turn that drive off.
2. Turn the computer off.
3. Insert the disc labelled "`BOOT:`" into drive 0 of your disc drive. Drive 0 is:
   - The right-hand internal drive in Model 226 and 236 computers
   - The left-hand drive in all HP external disc drives
   - The only flexible disc drive in combination Winchester/flexible disc drive units
4. Turn on all external disc drives that are connected to your computer. If a disc drive is not turned on while HP TechWriter is being booted, you will not be able to access it later unless you re-boot.
5. Turn your computer on. The computer locates the HP TechWriter system on the `BOOT:` disc and begins loading it.

   When the following display appears on the screen, HP TechWriter is ready for use. If the display fails to appear, or an error message is displayed instead, refer to the section called *Solving Booting Problems* in the *Pascal 3.0 User's Guide*.



```
New system date ?

  System date is          1-Mar-00
  Clock time is           00:00:23

  HP TechWriter           [Rev. 1.0]  1-Jul-84

  Total Available Memory 100000 bytes


  System  volume:   BOOT:
  Default volume:   BOOT:


Copyright 1983 Hewlett-Packard Company.
All rights are reserved.  Copying or other
reproduction of this program except for archival
purposes is prohibited without the prior
written consent of Hewlett-Packard Company.
```

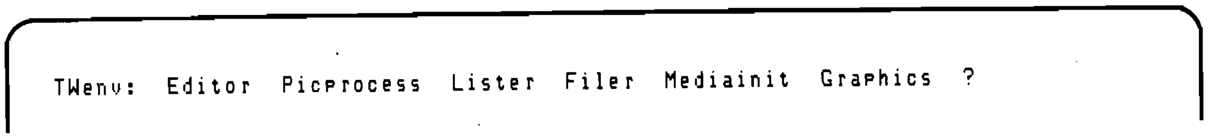
6. Enter the current date in the form shown: day, month, year. For example, March 14, 1985 is input as `14-Mar-85` (Return).
7. Enter the current time in the form: hour, minute, second. The time is in a 24-hour format. For example, 2:15:04 p.m. would be entered as `14:15:04` (Return).

The values entered here will be updated inside the computer as time goes on. The current date will be written into the directories of discs as you create or update your files. Thus, you can come back later and look to see when you last modified any particular file.

The date and time questions need not be answered if you do not wish to do so. If you just press (Return) in response to the questions, they will not change from those displayed, but will be updated from there.

The Command Line Now on Your Screen

After these two questions have been answered, the screen will be redrawn, and will then look like this:



```
TWenu:  Editor  Picprocess  Lister  Filer  Mediainit  Graphics  ?
```

From this point you press the capitalized letter of the subsystem you want to run. For example, if you wanted to use the Editor, you would press (E); if you wanted to use the Filer, you would press (F); if you wanted to use the Picture Processor, you would press (P), and so forth. When you press (?), the alternate prompt appears. All options available from the prompt are described in the following chapters.

Skip the following section and continue reading with the section *Important Information Either Way....*

If You Are Running Under the Pascal Operating System...

We assume here that your Pascal 3.0 system is running, and that the Main Command Prompt is displayed. If it is not, see the *Pascal 3.0 User's Guide* for instructions.

The Pascal editor (which comes with the Pascal system) is called `EDITOR`, and the HP TechWriter Editor is also called `EDITOR`. You cannot have two files of the same name in the same volume, so one will need to be changed.

Let's say that you want both the HP TechWriter Editor and the Pascal editor in a volume named `SYSTEM`. You need to rename the old Pascal editor, which is already in `SYSTEM:`, then copy the HP TechWriter Editor onto the same volume.

1. Load the `ACCESS:` disc and enter the Filer by pressing (F).
2. Invoke the Change command by pressing (C). This command allows you to change the name of a file. Type:

   ```
   SYSTEM:EDITOR (Return)
   PEDITOR (Return)
   ```
3. Load the `EDIT:` disc. Invoke the Filecopy command by pressing (F). Type:

   ```
   EDIT:EDITOR=(Return)
   SYSTEM:$(Return)
   ```

   This copies every file which starts with the letters `EDITOR` onto the new volume. Both the HP TechWriter Editor, and its configuration file are copied.

   Now you are ready to use the HP TechWriter Editor.

If you have other people working on the same system who want to use the old Pascal editor, either let them know the name has changed, or use the *What* command to tell your computer where to look for the HP TechWriter Editor. There is information on the What command in the *Pascal 3.0 User's Guide.*

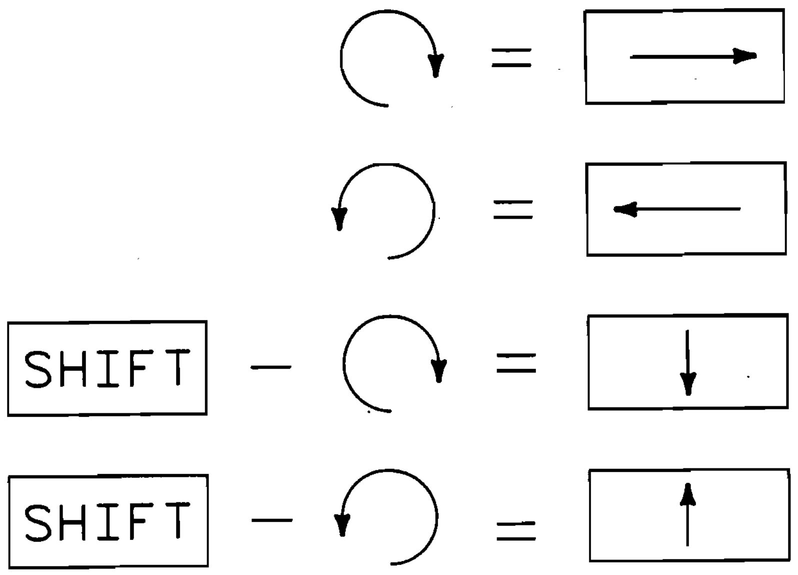
Important Information Either Way...

The HP TechWriter software is made for the Hewlett-Packard Series 200 computers. There are several different configurations available to Series 200 machines: different keyboards, different CRTs, different amounts of memory, etc. This manual may, therefore, describe an option that you don't have on your particular machine. *This will not affect the capabilities of the HP TechWriter programs in any way,* but the job may have to be done in a slightly different way.

The Knob

If you have a 98203A or 98203B keyboard, there is a "rotary pulse generator" – a "knob" – in the upper-left-hand corner of the keyboard. This knob can be rotated, making cursor movement faster than using the (▲), (▼), (◀), and (▶) keys. Clockwise rotation causes the same action as the (▶) key, counterclockwise rotation causes the same action as the (◀) key, shifted clockwise rotation causes the same action as the (▼) key, and shifted counterclockwise rotation causes the same action as the (▲) key (see picture).

Cursor Movement with the Knob



The Keycaps

Since there are several different keyboards available for Series 200 computers, there are some differences in individual key names that keyboards have. Usually, it will be easy to "translate" between one keyboard's key name and another's.

For example, whereas one keyboard may have an (EXECUTE) key, another keyboard has a (Select) key. To keep the differences straight, refer to the table below.

Keyboards

46020A	98203B	98203A
Return	ENTER	ENTER
Select	EXECUTE	EXEC
Caps	CAPS LOCK	CAPS
Clear line	CLR LN	CLR L
Clear display	CLR SCR	CLR S
Insert line	INS LN	INS L
Delete line	DEL LN	DEL L
Insert char	INS CHR	INS C
Delete char	DEL CHR	DEL C
Reset	RESET	RST
ANYCHAR[1]	ANY CHAR	ANY C
Shift-Select	SHIFT-EXECUTE	SHIFT-EXEC

This manual assumes you have a 46020A keyboard, and therefore talks about the Return key, the Select key, the Caps key, etc.

1 When a "keycap" is referenced without the key-shaped envelope around it, that function is accessed through the softkeys, numbered **f1** through **f8**. For further information on the softkeys for the HP46020A keyboard, see the *Pascal 3.0 User's Guide* in the section entitled *HP46020A Keyboard.*

Protecting Yourself

Chapter 2

Introduction

The discs you received with the HP TechWriter package have all been write-protected; that is, you cannot write any data to them. These discs should be copied and then stored away in a safe place (away from excessive heat, moisture, dust, and magnetic fields), to serve as backups in the event that your everyday discs wear out or become damaged.

This chapter describes three operations which must be done before you get into the next chapter. The three operations are:

1. Initializing a new disc, so that it can be used for information storage,
2. Copying files from the discs you received with the HP TechWriter package to spare discs. This is done so if something happens to the originals, you are not completely brought to a halt; you'll still have the backups.
3. *Using the freshly-made copies* of the discs you received from Hewlett-Packard, install your codeword, a security mechanism which discourages unauthorized copying.

Again, **you must make backup copies of the discs**. If you don't, not only will you be risking losing your software by having just one copy, but **examples in this manual will not work**. This is because: 1) the discs you get from Hewlett-Packard are write-protected, and some examples require you to write files, which cannot be done on write-protected discs. 2) Also, you need a codeword in your configuration file in order to write to files at all, and a codeword cannot be installed on a write-protected disc.

If you have already backed up all of the discs delivered with HP TechWriter, skip the next two sections and continue with the third section, *Software Security*.

Disc Initialization

Before your discs can be used, they must be prepared by a process called *initialization.* Your computer can not read from, write to, or even recognize a disc that has not been initialized. Initialization is done by the program `MEDIAINIT` (on the `ACCESS:` disc).

Note

If you want more information about disc initialization, or if you encounter an error in the following procedure please consult the *Preparing Your Discs* section of the *Pascal 3.0 User's Guide.*

The procedure below describes how to initialize a 3.5 or 5.25-inch disc in drive 0 (the drive you loaded the boot disc into) of your computer. Drive 0 is also called the *unit 3* drive.

You will need to initialize four flexible discs now, in order to make backup copies of your HP TechWriter discs.

Note

Only new discs should be initialized. Media initialization **destroys** all information on a disc. If you want to re-initialize a disc for any reason, be sure to save all of its valuable information on a separate disc before proceeding.

1. Make sure you have booted HP TechWriter and have answered the time and date prompts. The HP TechWriter command line should now be displayed at the top of your screen.
2. Insert the `ACCESS:` disc and load the `MEDIAINIT` program:
   a. If you are running the HP TechWriter system stand-alone (i.e., by itself), press (M) to load the program.
   b. If you are using the Pascal operating system, press (X). The system will respond:

```
Execute what file?
```

   Type `ACCESS:MEDIAINIT.` (Return) (note the period at the end of the file name), and the program will be executed.

3. No matter how you start the Mediainit program, when it starts running, the screen will be similar to this:

```
Mediainit [Rev 3.0  5/15/84]  17-Jul-84 15:10:55

     Copyright 1983 Hewlett-Packard Company.
            All rights reserved.

Volume ID?
```

   Load the disc to be initialized into the unit 3 drive (the drive you loaded the BOOT disc into when you booted your computer), close the door, and answer:

   `#3:` Return

   MEDIAINIT displays the device and the name of the existing directory (if any). It also gives this warning:

   ```
   Are you SURE you want to proceed ? (Y/N)
   ```

4. Make sure the disc about to be initialized is really the one you want, and answer yes by typing Y. (If you open the drive door, however, you will have to start the process over; since you opened the door, the computer doesn't know if the same disc is still there.)
5. The next prompt is:

   ```
   Interleave factor [1..15] (defaults to 1)?
   ```

   The computer will select a default that gives optimum performance for the disc drive you are using. Unless you have a specific reason to do otherwise, accept the default[1] by pressing Return.

6. The program will display the following message as initialization proceeds:

   ```
   Medium initialization in progress
   ```

   When initialization is complete, the program will display:

   ```
   Medium initialization completed
   ```

If Something Goes Wrong...

The program will display the following message:

```
Medium initialization aborted:
  medium initialization failed.
```

If this happens, your disc is damaged and you cannot use it.

---

1 "Default" is a word that means "the one used unless explicitly stated otherwise." Thus, "default interleave factor" is the one that is used unless another is specified.

7. The final step of initialization is volume zeroing. The program will display the following messages:

```
Volume zeroing in progress
```
then
```
Volume zeroing completed
```

8. The program will display the HP TechWriter main command prompt when disc initialization is complete.

   Your disc is now ready for use. A directory named *V3* (for "Volume 3", because this disc was initialized in *unit 3*) has been written on the disc with enough directory space to contain eighty files.

Use this procedure to initialize four new discs so you can make backup copies of the HP TechWriter discs.

Making Backup Copies of the HP TechWriter Discs

You can make backup copies of the HP TechWriter discs by copying all the files on one of the HP TechWriter discs to an empty disc. The empty disc must have been initialized as described in the preceding section.

Note

If you want more information about making backup copies of discs, or if you encounter an error in the following procedure please consult the *Backing Up Your System* section of the *Pascal 3.0 User's Guide.*

1. Make sure you have booted HP TechWriter and have answered the time and date prompts. The HP TechWriter command line should now be displayed at the top of your screen.
2. The command for copying files is contained in the *Filer*, one of the programs included in the HP TechWriter package. Make sure the `ACCESS:` disc is on-line, and then press (F). (The same technique is used to get into the Filer whether you are running HP TechWriter stand-alone or under Pascal.)
3. Now you're in the Filer. Insert the `EDIT:` disc, and press (F) for *Filecopy*. The Filer responds:

```
Filecopy what file ?
```

4. Type the name of the volume whose files you are copying, colon, equals sign. For example, if you are backing up the `EDIT:` disc, you would type `EDIT:=`. The equals sign means "all files." Now, press (Return). The computer responds:

```
Filecopy to what ?
```

5. Here, you insert the blank disc you're copying *to*. Then type the volume name of that disc. Say, for example, it is `V3:` – you would type `V3:$` (Return). (It would be `V3:` if you followed the instructions in the previous section.) The dollar sign means "use the same name." Put all together, the screen looks like this:

```
Filecopy what file? EDIT:=
Filecopy to what? V3:$
```

and it means: "Copy every (`=`) file from the `EDIT:` disc to the `V3:` disc, and give the `V3:` disc's files the same names (`$`) as the `EDIT:` disc's file names."

Note

This procedure assumes you are using two disk drives, one for the SOURCE disk and one for the DESTINATION. If you are using a single disc drive, the computer will display prompt messages asking you to insert SOURCE and DESTINATION discs at the appropriate time.

6. When the backup is complete the Filer prompt reappears. But before we backup the next disc, let's change the volume name `V3:` to one that is more descriptive. Press (C) to invoke the Change command.
7. Type:

   ```
   V3:(Return)
   EDIT:(Return)
   ```

   This changes the destination volume's name from `V3:` to `EDIT:`. We're done with these discs now. Remove them from the disc drives, and put a label on the outside of the new disc; write on it the same volume name as the source disc.
8. Repeat the routine described in steps 3 through 7 for all three of the remaining HP TechWriter discs. When backing up the `LIST:` disc, replace `EDIT:` in steps 3, 4, 5, and 7 with `LIST:`. When backing up the `BOOT:` disc, replace `EDIT:` in those steps with `BOOT:`. When backing up the `ACCESS:` disc, replace `EDIT:` with `ACCESS:`.

   If you work in an environment which is particularly dusty or otherwise hostile to magnetic media, you may want to make two backup copies of each disc.

   It is also advisable to use the same procedure as above for making copies of individual files created by the HP TechWriter programs. To copy a single file, follow the steps above, only substitute the name of the file to be copied for the equal sign. In this way, files can be copied one by one. For more information, see the *Filer Reference* chapter.
9. Now that you've made backup copies of the HP TechWriter discs, put the originals away in a safe place and use only the copies. Use the originals only to make more backups.
10. Press (Q) to leave the Filer.

If you want to write-protect your backup copies of the discs, do not do so until after you've installed the codeword. This is described in the next section.

Software Security

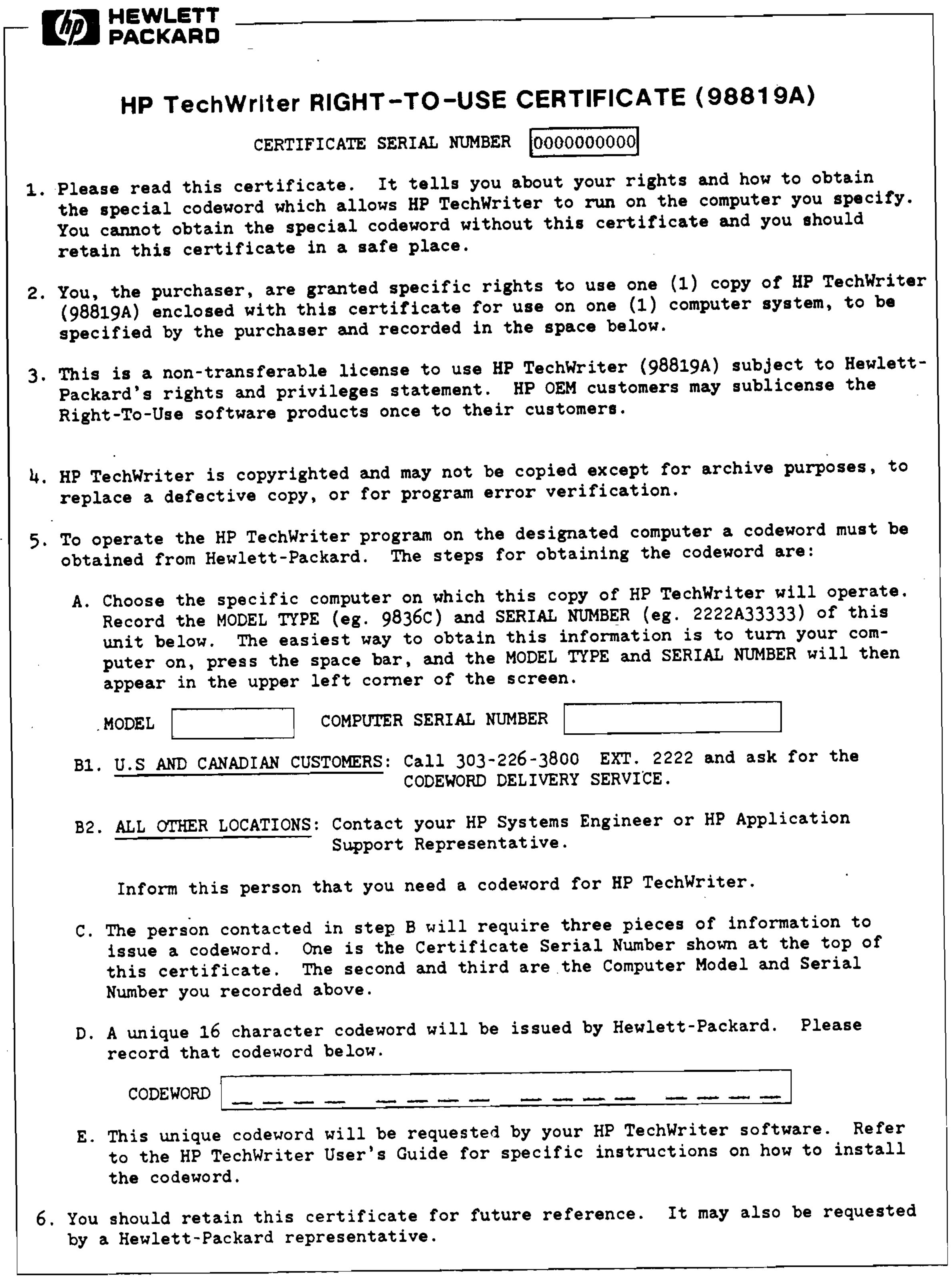
To protect the HP TechWriter programs from unauthorized copying, there is a special codeword that you will need to obtain from your Hewlett-Packard Sales Representative. If you do not have this codeword, you will not be able to save files created with the HP TechWriter Editor. The codeword allows the software to run on only one computer; a different codeword is necessary for running HP TechWriter on every individual machine.

On the following two pages are copies of the Right-To-Use and Right-To-Reproduce certificates. You received one of these with the HP TechWriter package. Follow the instructions outlined on the appropriate certificate to obtain your codeword, then read the next section, *Installing Your Codeword.*

Notice that the HP TechWriter discs supplied by Hewlett-Packard do not have a codeword installed on them (nor is there a way to install one since they are write-protected), so if you find it necessary to re-copy the `EDIT:` disc in the future, it will also be necessary to re-install the codeword.

You must have the codeword installed in order to save Editor work.

Sample Right-to-Use Certificate



HEWLETT PACKARD

HP TechWriter RIGHT-TO-USE CERTIFICATE (98819A)

CERTIFICATE SERIAL NUMBER 0000000000

1. Please read this certificate. It tells you about your rights and how to obtain the special codeword which allows HP TechWriter to run on the computer you specify. You cannot obtain the special codeword without this certificate and you should retain this certificate in a safe place.

2. You, the purchaser, are granted specific rights to use one (1) copy of HP TechWriter (98819A) enclosed with this certificate for use on one (1) computer system, to be specified by the purchaser and recorded in the space below.

3. This is a non-transferable license to use HP TechWriter (98819A) subject to Hewlett-Packard's rights and privileges statement. HP OEM customers may sublicense the Right-To-Use software products once to their customers.

4. HP TechWriter is copyrighted and may not be copied except for archive purposes, to replace a defective copy, or for program error verification.

5. To operate the HP TechWriter program on the designated computer a codeword must be obtained from Hewlett-Packard. The steps for obtaining the codeword are:

   A. Choose the specific computer on which this copy of HP TechWriter will operate. Record the MODEL TYPE (eg. 9836C) and SERIAL NUMBER (eg. 2222A33333) of this unit below. The easiest way to obtain this information is to turn your computer on, press the space bar, and the MODEL TYPE and SERIAL NUMBER will then appear in the upper left corner of the screen.

   MODEL ______ COMPUTER SERIAL NUMBER ______

   B1. U.S AND CANADIAN CUSTOMERS: Call 303-226-3800 EXT. 2222 and ask for the CODEWORD DELIVERY SERVICE.

   B2. ALL OTHER LOCATIONS: Contact your HP Systems Engineer or HP Application Support Representative.

   Inform this person that you need a codeword for HP TechWriter.

   C. The person contacted in step B will require three pieces of information to issue a codeword. One is the Certificate Serial Number shown at the top of this certificate. The second and third are the Computer Model and Serial Number you recorded above.

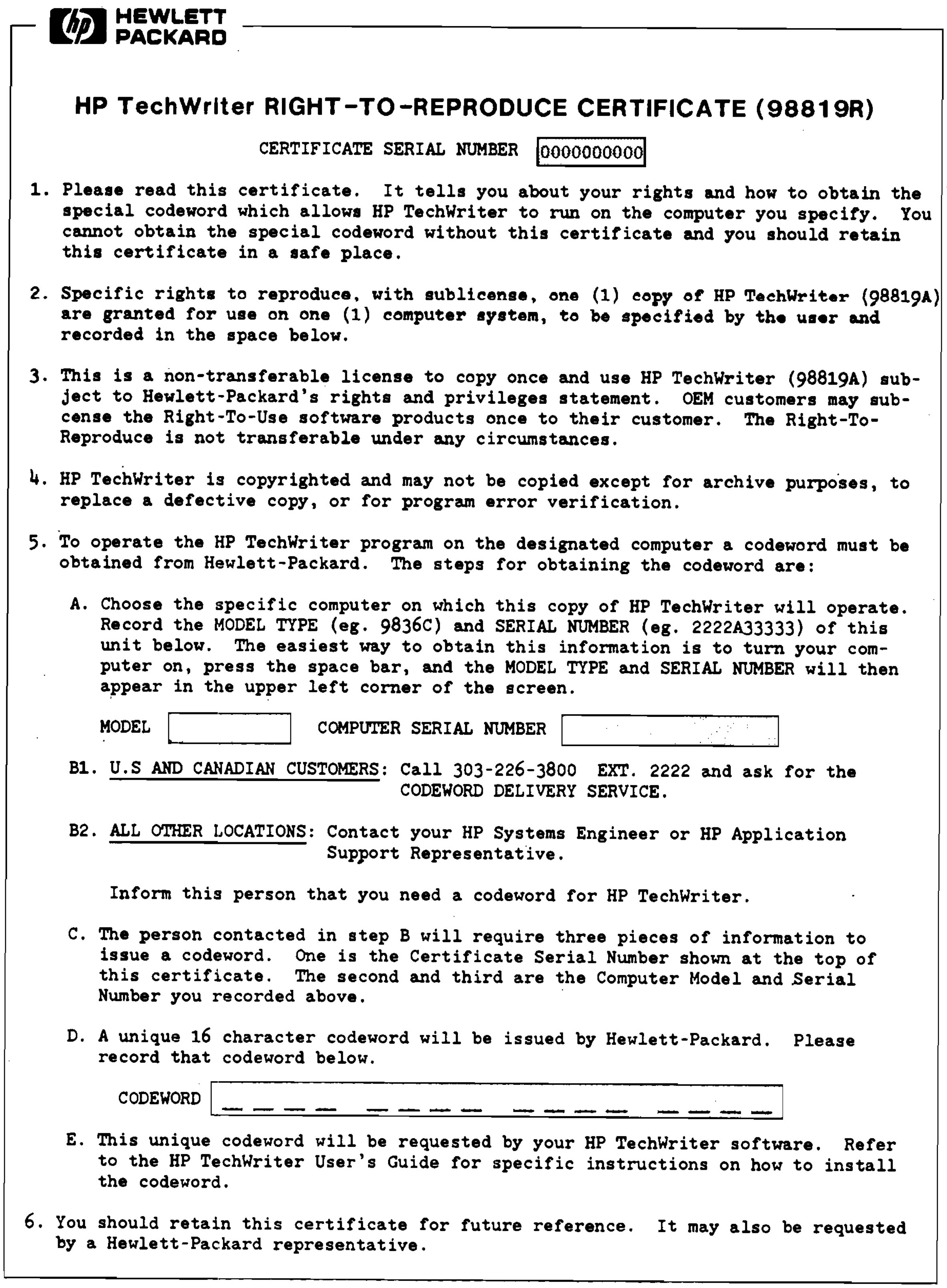
   D. A unique 16 character codeword will be issued by Hewlett-Packard. Please record that codeword below.

   CODEWORD _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

   E. This unique codeword will be requested by your HP TechWriter software. Refer to the HP TechWriter User's Guide for specific instructions on how to install the codeword.

6. You should retain this certificate for future reference. It may also be requested by a Hewlett-Packard representative.

Sample Right-to-Reproduce with Sublicense Certificate



HEWLETT PACKARD

HP TechWriter RIGHT-TO-REPRODUCE CERTIFICATE (98819R)

CERTIFICATE SERIAL NUMBER 0000000000

1. Please read this certificate. It tells you about your rights and how to obtain the special codeword which allows HP TechWriter to run on the computer you specify. You cannot obtain the special codeword without this certificate and you should retain this certificate in a safe place.

2. Specific rights to reproduce, with sublicense, one (1) copy of HP TechWriter (98819A) are granted for use on one (1) computer system, to be specified by the user and recorded in the space below.

3. This is a non-transferable license to copy once and use HP TechWriter (98819A) subject to Hewlett-Packard's rights and privileges statement. OEM customers may subcense the Right-To-Use software products once to their customer. The Right-To-Reproduce is not transferable under any circumstances.

4. HP TechWriter is copyrighted and may not be copied except for archive purposes, to replace a defective copy, or for program error verification.

5. To operate the HP TechWriter program on the designated computer a codeword must be obtained from Hewlett-Packard. The steps for obtaining the codeword are:

   A. Choose the specific computer on which this copy of HP TechWriter will operate. Record the MODEL TYPE (eg. 9836C) and SERIAL NUMBER (eg. 2222A33333) of this unit below. The easiest way to obtain this information is to turn your computer on, press the space bar, and the MODEL TYPE and SERIAL NUMBER will then appear in the upper left corner of the screen.

   MODEL ____ COMPUTER SERIAL NUMBER ____

   B1. U.S AND CANADIAN CUSTOMERS: Call 303-226-3800 EXT. 2222 and ask for the CODEWORD DELIVERY SERVICE.

   B2. ALL OTHER LOCATIONS: Contact your HP Systems Engineer or HP Application Support Representative.

   Inform this person that you need a codeword for HP TechWriter.

   C. The person contacted in step B will require three pieces of information to issue a codeword. One is the Certificate Serial Number shown at the top of this certificate. The second and third are the Computer Model and Serial Number you recorded above.

   D. A unique 16 character codeword will be issued by Hewlett-Packard. Please record that codeword below.

   CODEWORD _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

   E. This unique codeword will be requested by your HP TechWriter software. Refer to the HP TechWriter User's Guide for specific instructions on how to install the codeword.

6. You should retain this certificate for future reference. It may also be requested by a Hewlett-Packard representative.

Installing Your Codeword

The following procedure explains how to install the codeword you obtained from the Hewlett-Packard Codeword Delivery Service.

1. If your machine is not already in the HP TechWriter Environment or Pascal operating system, start it up.
   a. Insert the `BOOT:` disc into drive 0. If you have an external disc drive, make sure it is turned on.
   b. Turn on your machine.
   c. When the time and date prompts are displayed, press Return Return.
2. Remove the `BOOT:` disc and insert the `EDIT:` disc you made in the previous section. This disc *must not* be write-protected.
3. Press S. This tells the computer that you want to stream a file. The computer displays:

```
Stream what file ?
```

   A stream file is a file that tells the computer what to do much like the keyboard does.

4. Type `EDIT:MAKECODE` Return. The stream file called `EDIT:MAKECODE.TEXT` is executed, and you will be asked:

```
What is your codeword?
```

5. Type in the codeword which you received from the Hewlett-Packard Codeword Delivery Service, and press Return.
6. There will be a confirmation message asking you to verify that the codeword is correct:

```
Confirm your codeword.  Correct = <ret> or <ent>;  Incorrect = <STOP>
```

   If your codeword is correct, press Return. If it is not, press Stop, and start over at step 3.

   When you press Return, the stream file will enter your codeword in the configuration file. Your HP TechWriter Editor needs the codeword in order to allow it to save your work.

7. If you have installed your HP TechWriter Editor in another disc location, you *must* now copy the configuration file from the `EDIT:` disc to that location. If, for instance, you installed the Editor in a volume named `SYSTEM:`, you would use the following procedure:
   a. Load the `ACCESS:` disc and enter the Filer by pressing F.
   b. Load the `EDIT:` disc. Invoke the `Filecopy` command by pressing F.
   c. Type:

```
EDIT:EDITORC.ASC Return
SYSTEM:$ Return
```

This copies the configuration file, which now contains the codeword, into the same volume as the Editor.

From now on, the Editor will find the codeword on the configuration file.

Note

If for any reason the Editor cannot find the codeword in the configuration file, it will give you a chance to enter it from the keyboard. If you know how to use the Editor you can edit the configuration file yourself to install the codeword.

The Editor looks for the configuration file in the volume specified as the Editor's location in the `Whatfiles` command. The configuration file, `EDITORC.ASC`, *must* be placed in that volume, or the Editor will not allow you to save files unless you enter your codeword from the keyboard every time you start the Editor.

Now that you have given yourself an appreciable amount of insurance against losing the HP TechWriter files, let's go on to learn how to use the HP TechWriter programs.

Getting Started

Chapter 3

Introduction

There are three HP TechWriter programs that help you produce, print, and include illustrations in documents. They are the *Editor*, the *Lister*, and the *Picture Processor*. The Editor enables you to create, change, print, store and retrieve both document and program text. The Lister enables you to print multiple-file documents and to generate tables of contents. The Picture Processor converts plot files produced by graphics editors into picture files that the Editor and Lister can use.

This chapter introduces you to all three programs. By the end of this chapter you will be able to enter and modify text, print it, and save it in a file on your disc. You will also be able to use the Lister to print files, and to use the Picture Processor to convert plot files.

You can do a great deal more with the HP TechWriter programs than is discussed in this chapter. Chapter 4, *More About Editing*, continues the tutorial started in this chapter to teach you more about the power of the Editor commands. Chapter 5, *Fully Utilizing HP TechWriter*, discusses the commands you can use to tailor your printed document and the options you have for tailoring your system to your taste. If you have questions about any of the commands covered in any of these chapters, the three reference chapters later in the manual should answer them. Once you are more familiar with the HP TechWriter programs, you can use the reference sections for quick reference to the HP TechWriter commands.

In the back of the manual, there are several appendices which provide you with additional information about the HP TechWriter programs.

Note

All of the command prompts listed in this chapter assume an eighty-column screen. If your computer is a Model 226, which has a fifty-column screen, the prompts will be similar but abbreviated, and the text you type will fill the narrower display differently.

Editor Tutorial

This section introduces you to the HP TechWriter Editor, the program you use for creating and modifying text files. This section covers the commands and options you need to be able to write, print, and save a document. There are many extended versions of the commands and options available which allow you to more easily use some of the features and to tailor the appearance of your document. These are discussed further in the next two chapters, *More About Editing* and *Fully Utilizing HP TechWriter.*

The Editor has built-in reminders called prompts and uses single keystroke commands. It provides access to any part of a text file. It uses the cursor, the blinking underline symbol on the screen, to indicate where things will happen in the file. By moving the cursor, you can move rapidly through text files to read and edit your text.

Entering the HP TechWriter Editor

1. Put the `EDIT:` disc into an on-line disc drive.
2. Press E from the main command level to run the Editor. If you receive the missing codeword alarm, you should press Stop and follow the procedure described in the *Installing Your Codeword* section of the last chapter. Otherwise, proceed to step 3.
3. When you enter the Editor, the computer displays the following prompt (assuming the proper codeword has been installed):

```
HP TechWriter EDITOR [Rev. 1.0]  1-Jul-84

Copyright 1983 Hewlett-Packard Company.
         All rights reserved.

No workfile found.
File? (<ret> for new file, <stop> exits)
:
```

Note

If this is not displayed on your screen, but the Editor command line (it starts with `>TWedit`) is displayed, you have what is called a *workfile.* A workfile is a special file which the Editor looks for first. If it exists, it is loaded; if it does not, the Editor asks you for a file name.

To use this tutorial, you must remove the workfile. Exit the Editor (press Q E), and go into the Filer (press F). If you want to save the file, press S. Then press N (for New workfile), to get rid of the old workfile. If you do not want to save the file, just press N to purge the old workfile. (In this case, you will get a message:

```
Throw away current workfile ? (Y/N)
```

Just press Y for yes.) Now press Q to quit the Filer, and E to re-enter the Editor. Now the display should look like the above.

The display above tells you that you are entering the Editor and requests a file name. Respond by pressing the Return key to instruct the Editor to create a new file for your use. Now that you've entered the Editor successfully, you need to learn the general form that the Editor commands take. The next section describes this.

The Anatomy of a Command

When you pressed Return in the previous section, the screen cleared and displayed the Editor prompt on the top line:

```
>TWedit:  AdJust  Copy  Delete  Find  Insert  Jump  Replace  Quit  Xchange  ?
```

You are now in the HP TechWriter text Editor with a new file. The file currently in the Editor is empty, of course, since you haven't inserted any text.

All of the commands in the Editor are initiated by typing a single key. The key you press corresponds to the capitalized letter in the command name. The names of all the commands are shown in the Editor prompt, which is the line starting with `>TWedit:` at the top of your screen. It does not matter whether the character you type is upper case or lower case.

The Editor prompt shows the most common commands used in the Editor. This is called a "prompt" because it prompts you to take some action, that is, to give the Editor a command. The prompt is a line of commands which can be used at this point. This line of available commands – the "command line" – is always displayed unless you are currently using one of those commands.

The first character of the command line (">" or "<") indicates the direction the cursor will move through the text when you press Tab, Return or the space bar. When the ">" character is displayed, the cursor will move forward in the text. It should be in this state now.

1. Press <. Now the direction indicator is "<", indicating that the cursor will move backwards in the text when you press Tab, Return or the space bar. Pressing any of the keys >, ., or + will set forward direction, while pressing any of the keys <, ,, or - will set reverse direction. Experiment with these keys to see how they can be interchanged and pressed in any order.

   The cursor direction character also indicates the direction that searches take place in the Find and Replace commands, and also the direction of the Page and Margin commands. These may be unfamiliar terms right now, but we'll cover them in due time.

2. The command line shows only a partial list of commands available in the Editor. To see the rest of the commands, type (?). The screen shows the Editor's alternate prompt:

```
>TWedit:  Zap  List  Margin  Page  Set  Verify  Help  ?        [1.0]
```

   This "alternate" prompt (which merely shows you the remainder of the commands available) also shows the revision number of the Editor enclosed in brackets. Type (?) again and the main Editor command line reappears.

3. Let's examine a typical command to see how it works. Press (I) to initiate the Insert command. The screen responds with the following prompt:

```
>Insert: <text> <bs> <clr ln> [<sel> inserts, <sh-sel> aborts]
```

   This command, like many others, has keywords within angular brackets. These keywords define what characters are legal now. `<text>` says you can insert text. Type some characters. They appear on the screen at the cursor position, and the cursor moves to the next position.

4. The `<bs>` says that you can back up to erase characters already typed. `<bs>` can be done by (**Backspace**), or (◀). Try both of these methods of backing up.

   Try to back up beyond the point at which you started the insertion. The computer will beep, and you will get the message:

```
>ERROR: Can't back up, <space> continues.
```

5. Press the space bar, type a few characters, and then press (**Clear line**). `<clr ln>` causes the text inserted thus far to be erased back to the end of the previous line, if there is a previous line on this insert. If there is not (and there isn't in this case), the computer will beep and display the same message:

```
>ERROR: Can't back up, <space> continues.
```

6. To clear the error condition, press the space bar, as the message indicates.

7. Next, while holding down the (**Shift**) key, press (**Select**) (this is indicated by (**Shift**)-(**Select**)). This causes the Editor to abort the insertion, and the main prompt line comes back. We'll see in the next section, where we create some *real* text, how to accept the text you type.

A Sample Editor Session

We encourage you to actually do the following examples on your computer. Actually doing the examples will help you learn faster than just reading about them.

Creating Your First Paragraph

The most direct way to generate text is with the Insert command.

1. Press I, as you did in the previous section. The insert prompt comes up:

```
>Insert: <text> <bs> <clr ln>  [<sel> inserts, <sh-sel> aborts]
```

Type the words "The German scientist". What probably appeared on your screen (unless you're getting ahead of us) was "tHE gERMAN SCIENTIST". The reason for the strange appearance of the words is this: When you first turn on your computer, it wakes up with *caps lock* on. When *caps lock* is on, all the letters of the alphabet are capitals (uppercase) unless you hold the Shift key down while pressing the letter. When *caps lock* is off, all the letters of the alphabet are small (lower case) unless you hold the Shift key down while pressing the letter.

Pressing the Caps key causes the *caps lock* state to alternate between "on" and "off."

In the lower right corner of the screen, there is either a caret (^) or a lowercase V (v). The caret "points up," indicating upper case (*caps lock* is on), and the lowercase V "points down," indicating lower case (*caps lock* is off).

2. Press the Caps key a few times and notice that the "^" and the "v" alternate. Most typing will be done with the "v" displayed; that is, letters will be lowercase unless you hold down the Shift key while pressing a letter.

   Now you know why tHE gERMAN SCIENTIST appeared instead of The German scientist.

3. Press Backspace enough times that all the text disappears, press the Caps key so that a "v" is in the lower right corner of the screen.
4. Enter the text as shown in the example below, including the positions of the ends of the lines. Press Return to cause a new line to begin. If you make a mistake, use Backspace to move the cursor backward and then type the correction. As mentioned above, most command prompts in the Editor show actual key options in the form of a key abbreviation shown in angular brackets.

   The word "definite" is misspelled in the text; leave it that way for now.

```
>Insert: <text> <bs> <clr ln> [<sel> inserts, <sh-sel> aborts]

 The German scientist
 Friedrich Mohs (1773-1839)
 developed a scale
 of hardness to assign a definate figure to the values he obtained in
 determining mineral hardness.
```

Now press Select to accept the insertion. Notice that the insert prompt disappears and the main Editor prompt returns.

You may have noticed that as you were typing the long line, the Editor beeped as you neared the right edge of the screen. This was a signal for you to press Return, which moves the cursor to the next line of the screen. While it's not a particularly heavy burden to press Return at the end of every line, it would be nice if the Editor noticed, and automatically moved the cursor. Not only that, but if a word is not going to fit on the current line, the Editor should move that word to the beginning of the next line. The Editor can do both of these things; we just need to instruct it to do so.

Word-Wrap (Document Mode)

"Word-wrap" means that if a word will not fit on the current line, the computer will automatically "wrap it around" (move it) to the next line. Before entering the next sentence, let's tell the computer to word-wrap.

1. Press the S key, and the computer assumes you want to set something in the "environment." The environment is a set of conditions under which the Editor operates. There are five options available when you press the S key:

```
>Set:  Env Mark Prog Doc  [<sh-sel> to leave]
```

   The ">" merely notes which way the cursor will move (see the *Anatomy of a Command* section earlier in this chapter); it cannot be changed at this point.

2. Press D. This sets Document mode, the mode in which the Editor does word-wrap. The screen will completely change to show the settings of the global environment. The options available from this environment will be covered in detail later.

3. For the moment, just press a space. This exits the environment menu and returns you to the main Editor prompt and the display showing your text.

   The mode that you just set remains in effect for the rest of the editing session, unless you change it again. It is even stored in the file. When you come back to this file next week, the environment will be exactly as you left it.

Automatic Margining During Insertion in Document Mode

Now you realize you "forgot" to put in a couple of words in the sentence you typed.

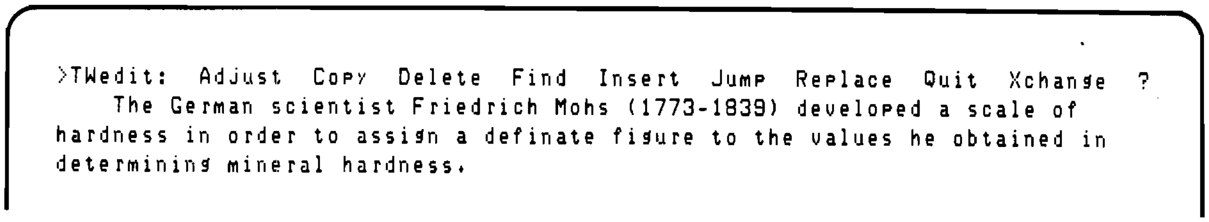
1. Use the arrow keys to move the cursor to the space after the word "hardness" in the phrase "scale of hardness".

2. Press I to insert, and you'll notice that the remainder of the line (right of the cursor) moves as far as it can to the right. This makes room for you to insert something. If you insert more text than will fit in the space opened up, the remainder of the text on that line moves down one line, and the text below that is erased from the screen. When you eventually press Select, the erased text will be redisplayed.

   The text we want to insert is "` in order`". Note that there is a space before the new word "in".

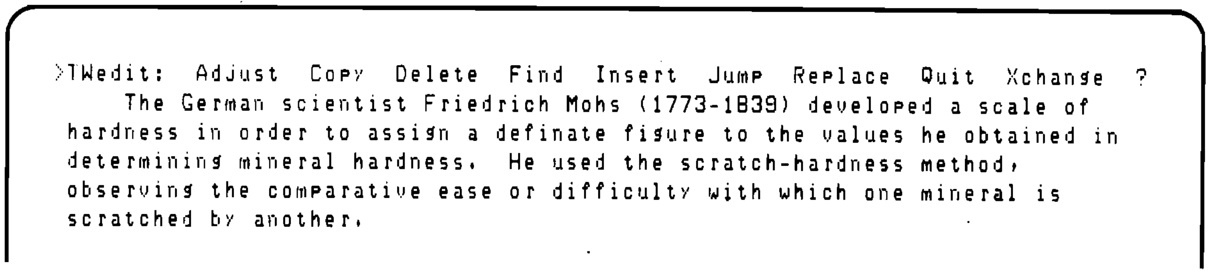
3. Type " `in order`" followed by Select. When you pressed Select, accepting the insertion, you may have been surprised to see the word arrangement on the screen drastically changed. The sentence was not changed, nor was its meaning, but the short lines were combined into longer lines to make the sentence "look better." This process of making the margins look better is called *margining*, and it was enabled when you pressed S D above, setting "Document mode."

   Your screen should now look like this:



```
>TWedit:  AdJust  Copy  Delete  Find  Insert  Jump  Replace  Quit  Xchange  ?
    The German scientist Friedrich Mohs (1773-1839) developed a scale of
hardness in order to assign a definate figure to the values he obtained in
determining mineral hardness.
```

4. Now use the arrow keys to move the cursor to the end of the sentence. Press I to begin inserting text, then type in the second sentence as shown below, ending with Select. Since Document mode is set, words extending over the end of the line will be moved to the next line, and the paragraph will automatically be margined when you press Select.



```
>TWedit:  AdJust  Copy  Delete  Find  Insert  Jump  Replace  Quit  Xchange  ?
    The German scientist Friedrich Mohs (1773-1839) developed a scale of
hardness in order to assign a definate figure to the values he obtained in
determining mineral hardness.  He used the scratch-hardness method,
observing the comparative ease or difficulty with which one mineral is
scratched by another.
```

   Now the Editor automatically takes care of the end-of-line processing. That is, you don't have to press Return to cause the cursor to go to the beginning of the next line. As you were typing, the Editor noticed when you went past the right margin, and took the entire final word from that line and placed it on the next line.

   The display above shows what your screen should look like after the first two sentences are entered and Select has been pressed.

Next, you will learn how to save this text file on the disc and then return to edit the text some more.

Saving Your File and Returning to the Editor

When editing, it is a good idea to save your file periodically to protect your work. You protect your work by telling the Editor to place it onto a storage medium (a disc). Unlike memory, where your text has been up to this point, a disc retains information even when you turn your machine off.

1. Press Q to initiate the Quit command. The screen clears and displays:

```
>Quit:
     Update the workfile and leave
     Exit without updating
     Return to the editor without updating
     Write to a file name and return
```

2. Press W (for `Write to a file name and return`) and the screen displays:

```
>Quit:
Name of output file (<ret> to return) -->
```

   The prompt is requesting a file name. The file you name will be placed onto the default volume unless you specify another volume. For example, if you want a file named `Mineral` to be placed on volume `EDIT:`, you would type `EDIT:Mineral` in response to the file name question.

   When specifying volume and/or file names, *case is significant* – upper case letters are not the same as lower case letters. That is, the file name `Mineral` is different from `mineraL`, which is different from `mineral`, etc.

3. Respond to the file name question by typing `EDIT:Mineral` and pressing Return. The Editor stores the file currently in memory into a file called `Mineral.TEXT` which is on the volume called `EDIT:`. Volume `EDIT:` must be on-line when you tell the Editor to put something there!

   The Editor stores the file as type TEXT. This means that the file name will have `.TEXT` appended to it, and the environment will be saved along with the file.

   The screen now displays:

```
>Quit:
Writing..
Your file is 316 bytes long.
Exit from or Return to the editor?
```

The exact number of bytes may differ from what is indicated in the line above, depending on whether you put any blank lines into your file, followed the period with one blank or two, etc., but it should be quite close.

4. Now press R to Return to the editor. The screen fills with your text and the cursor is positioned where it was when you initiated the Quit command.

Changing Text

If you are like some people, it grates on your nerves to have to tolerate the intentionally misspelled word in our example paragraph ("definite" was spelled "definate"). So, we'll hurry up and correct it.

1. Use the cursor-control keys (the arrow keys) to move the cursor to the "a" in the misspelled "definate", then press X. This invokes the "eXchange" command, which exchanges characters one-for-one until the end of line. The eXchange prompt looks like the following:

```
>Xchange: <text> <bs>  [<sel> changes, <sh-sel> aborts]
```

2. Press "i" (make sure it's a lower case "i"; if it isn't, press Backspace and try again) followed by Select, and the misspelling is corrected.

   While Xchanging characters is sufficient for small modifications, when you have larger ones, the Replace command can be better.

3. Move the cursor to the beginning of the file, and then press R, which stands for Replace. The following prompt appears:

```
>Repl[1]: T V <target><repl>=>
```

   We'll deal with the meanings of "T" and "V" later. For now, type:

   ```
   /scientist//mineralogist/
   ```

   This tells the Editor to replace the next occurrence of the word "scientist" with the word "mineralogist". The slashes are delimiters – they inform the Editor of the beginning and ending points of the string to be replaced (the `<target>`), and the string to replace it (the `<repl>`, or replacement).

During a Replace operation in Document mode, margining is *not* automatically done, as it is when inserting.

Your text should now look like this:

```
>TWedit:  Adjust  Copy  Delete  Find  Insert  Jump  Replace  Quit  Xchange  ?
    The German mineralogist Friedrich Mohs (1773-1839) developed a scale of
 hardness in order to assign a definite figure to the values he obtained in
 determining mineral hardness.  He used the scratch-hardness method,
 observing the comparative ease or difficulty with which one mineral is
 scratched by another.
```

Deleting and Recovering Text

The HP TechWriter Editor also allows you to delete text.

1. Move the cursor to the "m" in "mineralogist" and press D to initiate the Delete operation. The Editor prompts you with the following message:

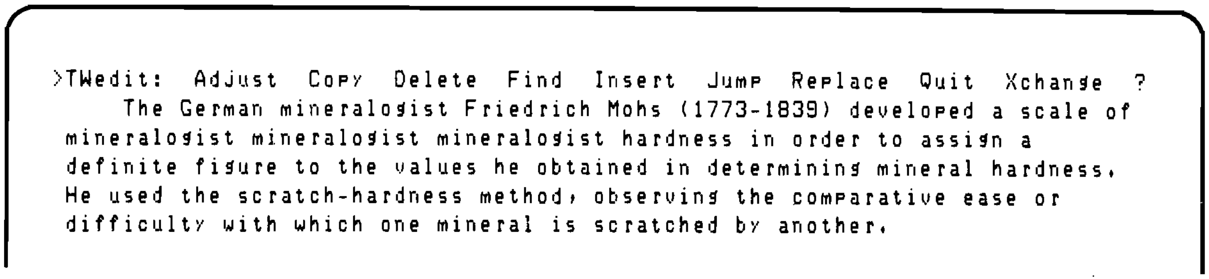
```
>Delete: < > <arrow keys>      [<sel> deletes, <sh-sel> aborts]
```

   This prompt tells you that you can delete text by pressing the "arrow keys." The phrase "arrow keys," in this context, is not limited to ◀, ▶, ▲, and ▼, but also includes the space bar, Return, Backspace, and Tab. In short, "arrow keys" here implies any key which moves the cursor. Text which has been deleted off the screen can be "undeleted"; you merely move the cursor back toward the location the cursor was at when the Delete command was entered. This will work as long as you haven't pressed Select, which tells the Editor that you really want to delete the tentatively-removed text. As the prompt implies, pressing Select accepts the deletion, and Shift-Select aborts the deletion, leaving your text unchanged.

   The cursor should now be under the "m" in "mineralogist," and the Delete command should have been invoked.

2. Now press the space bar to move the cursor to the "F" in "Friedrich." You will notice that the text was deleted as you went. When you press Select, you actually delete the text, and "mineralogist" is gone. However, you can still recover if you need to, because even if you've pressed Select, you can get your text back by using the Copy Buffer command.
3. The "buffer" is a special place which holds the most recently deleted text. To copy the buffer back into your file, press C B and "mineralogist" comes back. You can move the cursor before you press C B, which means you can move text to different places. Copying the buffer back into your text does not empty the buffer, either, so you can duplicate text by pressing C B several times.
4. Press Return to go to beginning of the next line, and press C B C B C B. This copies the contents of the copy buffer into your file three times.

5. Press M to margin the paragraph. "Margining the paragraph" means that the editor packs words as tightly as possible, which makes the margins look as nice as possible. Now your text should look like this:



```
>TWedit:  Adjust  Copy  Delete  Find  Insert  Jump  Replace  Quit  Xchange  ?
     The German mineralogist Friedrich Mohs (1773-1839) developed a scale of
 mineralogist mineralogist mineralogist hardness in order to assign a
 definite figure to the values he obtained in determining mineral hardness.
 He used the scratch-hardness method, observing the comparative ease or
 difficulty with which one mineral is scratched by another.
```

6. Since we don't *really* want "mineralogist" to appear three extra times, let's delete them. Your cursor should be under the "m" of the first superfluous "mineralogist." Press 3 R (to replace the next three occurrences) to invoke the replace command. Type:

   ```
   /mineralogist///
   ```

   What you actually did was "Replace the next three occurrences of 'mineralogist' with nothing." The replacement string is that between the last two delimiters – nothing in this case. Effectively, this is a delete. But notice one thing. When you copied from the copy buffer, you got "mineralogist" followed by a space. When you replaced the three occurrences of "mineralogist" with nothing, you did *not* delete the space also. So, there are three extra spaces left over in the text. We can easily fix that by margining the paragraph again.

7. Do so by pressing M and notice what the paragraph looks like.

8. Now margin the paragraph again, and you'll notice that nothing changed; the margins are already as smooth as they can get, given the current text and environment parameters. Margining a paragraph immediately after it was margined does not change anything.

Placing Drawings In Your File

One of the most exciting features of your HP TechWriter Editor is its ability to place pictures into your document. A "picture file" is a file containing the information necessary for a plotter to draw a picture. We have provided a processed picture file on your `LIST:` disc which you can load right now. Let's find out how to get pictures into our file.

1. Place the `LIST:` disc into a disc drive, and close the door. Now, in the text file, we'll insert the commands to access the picture file.

2. Move the cursor to the end of the file. Press I to invoke the Insert command, and press Return Return followed by:

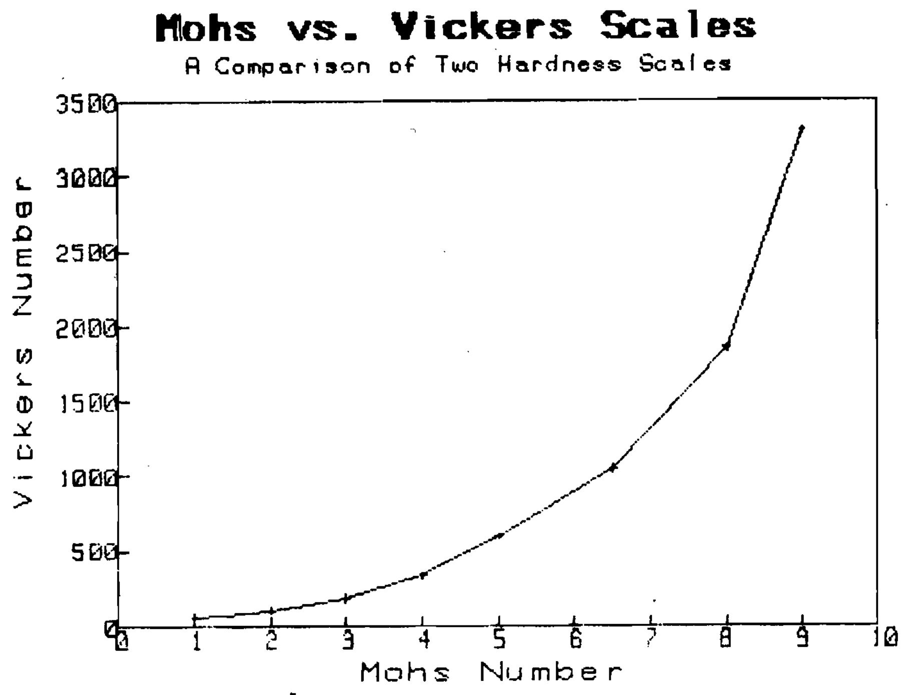
   ```
   ^^D F=LIST:mohs
   ```

3. Press Return seventeen times, to insert sixteen blank lines.

4. Type `^^`. This tells the Editor where the bottom of the picture is supposed to be.

5. Finally, press (Select). This sequence of commands tells the Editor to insert a drawing here (^^D). The file in which the picture resides (F=) is mohs, and that file is on the volume LIST:. Draw the picture in the space between the drawing command (the ^^D) and the next document command (the ^^). If you don't understand all this, don't be concerned; we will cover it in much more detail later. Right now, you probably just want to see a picture.
6. After inserting all the correct commands, still nothing happens. The reason for this is that the environment (that set of conditions under which the Editor operates) thinks you don't want pictures to be drawn. To tell it otherwise, press (S) (E) (D) (T), and press the space bar. This told the Editor: *S*et the *E*nvironment: *D*rawing *T*rue. Pressing the space bar exited the environment.

   Now, move the cursor up until the following picture appears on your screen.



7. Note that after the picture is drawn, you can scroll the picture as well as the text. Press the (▲), (▼) or (Return) keys until the picture scrolls off the screen, then scroll it back on. Note how it is erased as soon as any part of it moves off the screen, and it is redrawn as soon as it can completely fit on the screen again.

Listing Your Document from the Editor

Assuming that you have a printer connected to your computer and turned on, it is very simple to get a listing of your document. Not only can the text in a document be listed, the pictures can also be sent to the printer. The following Hewlett-Packard printers are supported for text and graphics: 2631G, 2671G, 2673A, 9876A, ThinkJet, 2932A, 2933A, 2934A, and 82906A. The following Hewlett-Packard printers are supported for text only: 2631B and 2671A.

1. Press L from the main command level. The List menu comes up:

```
>List setup: {options}; <L lists>, <sel> or <sp> leaves

Number of lines per page:  60                 Include Tags:         True
First page number:         1                     "    Commands:     False
Top lines skipped:         0
                                              Draw figures:         True
                                              Continuous form:      True
Printer type:              2934A
Output spool file - PRINTER:
```

2. If your printer is not a 2934A, press P. If you have one of the supported printers, enter the appropriate printer type, followed by Return. If you have a printer that is not one of those supported, type "UNKNOWN" followed by Return and the computer will do its best.
3. If you have a non-graphics printer or any non-supported printer, you should set Draw figures False in the *List* environment (press D F).
4. The Output spool file parameter specifies which printer is to be the destination of the listing. This parameter has a default value of PRINTER:, which means to use the *local* printer (a printer directly connected to your computer). If you do not have a local printer, press O and enter the appropriate printer specification. For example, if you are on a Shared Resource Manager (SRM) system, you might enter something like #5:/PRINTER/MYFILE.ASC.

   We'll get into the meanings of the other options later.
5. When all options are set as you want them, just press L again, and the file currently in the Editor will be sent to the file, and the prompt

```
>Listing...
```

will be displayed. After the listing is completed, the main List prompt returns. Press the space bar to leave the List menu. That's all there is to it.

Archiving Your File and Exiting the Editor

Now we are almost done with the Sample Editor Session, so let's leave the Editor. We've changed the file, so we'll exit the Editor after storing a new copy of the file. This time we'll use an option that saves, or *archives*, a copy of what we stored last time.

1. Make sure the `EDIT:` disc is in the disc drive, and the door is closed.
2. Press Q, and the Quit prompt comes up:

```
>Quit:
    Update the workfile and leave
    Exit without updating
    Return to the editor without updating
    Write to a file name and return
    Archive file, write new file EDIT:Mineral.TEXT
    Save as file new file EDIT:Mineral.TEXT
    Overwrite as file EDIT:Mineral.TEXT
```

3. There are three more options available this time, because the Editor knows what the file name could be. For now, just press A, to `Archive file, write new file EDIT:Mineral.TEXT`. The Editor responds:

```
>Quit:
Writing..
Your file is 394 bytes long.
Exit from or Return to the editor?
```

4. When a file is archived, if a file already exists with the name indicated it will have an `@` appended to its name. Then, the current contents of the Editor are written to a new file which is given the file name shown. In this example, `EDIT:Mineral.TEXT` (which was made the first time we saved a file) gets renamed to `EDIT:Mineral@.TEXT`, and the current Editor contents are stored in `EDIT:Mineral.TEXT`.

   Now, press E, and you are out of the Editor.

This is the end of the Editor Tutorial. You should now be able to write a simple memo, incorporate an already-processed picture, save the memo on disc, and print it on your printer.

The next section introduces the HP TechWriter Lister program. The Lister enables you to print multiple files as a document.

Lister Tutorial

While listing a single file from the Editor is usually satisfactory, there may be times when you want several files to be printed together as a single document. This can be difficult or even impossible to do with the Editor alone.

The *Lister* is another program in the HP TechWriter package. The Lister provides the capability of printing many files in one document. It can store the list of file names required to create the multi-file document, so you don't have to reenter it every time you want the large document printed. The Lister also can create tables of contents.

Let's try a few examples which show how to use the Lister program. If you have questions about the commands covered, there is more information about the Lister in the *Lister Reference* chapter of this manual.

Entering the HP TechWriter Lister

The first thing you will need to do is to start the Lister program.

1. Load the `LIST:` disc into a disc drive and shut the disc drive door.
2. Enter the Lister by pressing $\boxed{\text{L}}$ from the main command prompt of the HP TechWriter Environment, or by executing $\boxed{\text{X}}$ `LIST:LISTER.` from the Pascal Operating System's main command line.

   The Lister's prompts and commands are very similar to the Editor's. When you run the Lister, the first thing it displays is this:

```
HP TechWriter LISTER [Rev. 1.0]  1-Jul-84

Copyright 1983 Hewlett-Packard Company.
        All rights reserved.

List file? (<ret> for new file, <stop> exits)
:
```

   The Lister is asking you for the name of a list file. A list file is a file which contains a file pool and the values you set for the listing parameters. A file pool is the list of the file names you want to have printed.

A list file is made by the Lister, so the first time you run the Lister, you will not have a list file.

A Sample Lister Session

1. Press `Return` now, since you don't have a list file to recall. The screen is erased, and the following is displayed:

```
TWlister:  Add files  List  Quit  Insert  Delete  Replace  <crsup/down> ?

                                  File name
                                  ---------
```

   The Lister is waiting for you to enter some file names; most of the rest of the options don't make sense unless there is at least one file name defined.

2. Press `A` for `Add files`. The screen then displays:

```
Add file: Filename <ret>; empty line or <sel> leaves

                                  File name
                                  ---------
 1  File name:
```

   The Lister is now ready to have you enter a file name.

3. Insert the `EDIT:` disc, and type the following file names. At the end of each file name, press `Return`.

```
1  File name: EDIT:Mineral
2  File name: EDIT:Mineral@
3  File name:
```

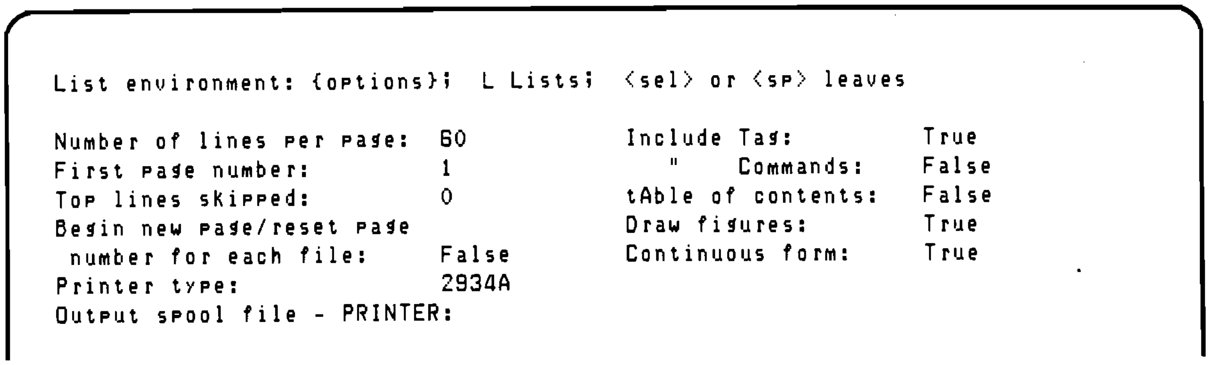
   When the time comes to list these files the Lister will automatically append `.TEXT` to file names just like Editor does. If the file does not have a `.TEXT` suffix, you must type the whole file name and add a period at the end of the file name.

4. Now that you have typed the file names as above, press `Return` *without* typing a file name. This means that you have finished adding file names. The screen will be reprinted as shown below:

```
1: EDIT:Mineral
2: EDIT:Mineral@
```

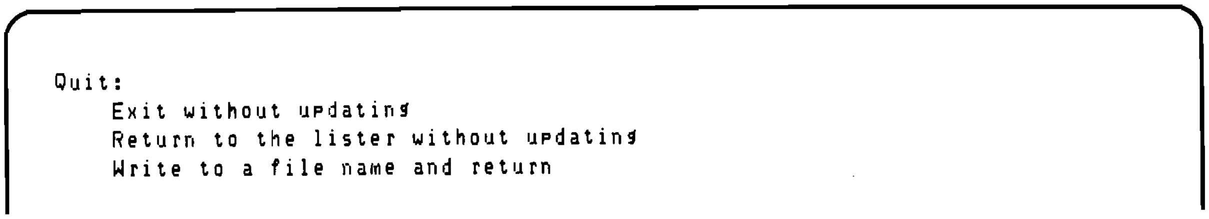
5. Now that there are some file names on the screen, press `▲` (or `◀` or `Backspace`) and `▼` (or `▶` or `Return`) a few times. The cursor moves up and down as you expect.

6. Now, press L. The list menu is displayed:



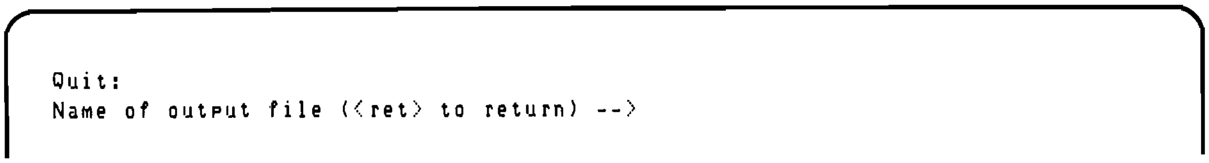
```
List environment: {options}; L Lists; <sel> or <sp> leaves

Number of lines per page:  60        Include Tag:          True
First page number:         1            "   Commands:      False
Top lines skipped:         0         tAble of contents:    False
Begin new page/reset page            Draw figures:         True
 number for each file:     False     Continuous form:      True
Printer type:              2934A
Output spool file - PRINTER:
```

7. If the printer parameters are not already correct, set them:
   - If your printer is not an HP 2934A, press P and specify the type of printer you have.
   - If you do not have a local printer, press O and enter the name of the file to which you want the printing sent.
   - If your printer does not support graphics, press D F to turn off printing of pictures.
8. Press L to start the listing. The files you created in the tutorials, earlier in the chapter, are first checked to ensure that they are available for listing, and are then printed. Notice that the Lister does not start a new page between the files.
9. After the printing stops, press B T to set `Begin new page/reset page number for each file` to `True`.
10. Press L to list the files again. This time, when the second file starts, it starts on a new page, and page numbering start over at page one.
11. Press the space bar to leave the listing environment.
12. Here is a good place to store a list file.
    a. Press Q. The Lister displays:



```
Quit:
    Exit without updating
    Return to the lister without updating
    Write to a file name and return
```

    b. Press W. The Lister responds:



```
Quit:
Name of output file (<ret> to return) -->
```

c. Type `EDIT:MyDoc` (Return). The Lister writes the current file pool and its listing parameters onto the file `EDIT:MyDoc.LS`. If you don't want the `.LS` suffix, you can put a period at the end of the file name. However, if you do so, you must remember that that file is a Lister file; there is no longer a `.LS` suffix appended to the file name to jog your memory.

13. Now the Lister's screen looks like this:

```
Quit:
Writing..
Exit from or Return to the Lister?
```

14. Press (E) to exit the Lister.

Modifying the File Pool

We have just created a file pool, modified listing parameters, listed the files, and stored the information on a list file. Now let's re-enter the Lister, call up the list file we stored previously, and look more at the listing options.

1. Re-enter the Lister the same way you entered it before. When the prompt comes up asking for a file name, type `EDIT:MyDoc` (Return). The Lister finds the list file, loads it, and displays:

```
1: EDIT:Mineral
2: EDIT:Mineral@
```

2. Press (A) to tell the Lister you want to add more files. Add the following file name: `#5:/Desk/LastFile.`. Note the period at the end of the file name. This inhibits the automatic appending of the `.TEXT` to the file name. The slashes in the new file name indicate that this file is on an SRM system (if you don't have one, don't worry about it; we aren't really going to list this file anyway).

3. Press (Return) (Return) at the end of the file name. The first (Return) indicates the end of the file name, and the second one indicates a null entry. This stops the Add mode. Your screen now should look like this:

```
1: EDIT:Mineral
2: EDIT:Mineral@
3: ../Desk/LastFile.
```

4. Now you decide you would like to add a new file name right before `EDIT:Mineral@`. Inserting takes place immediately *before* the file indicated by the cursor at the point where Insert is invoked.

   Move the cursor to file number two: `EDIT:Mineral@`, and press (I). The screen displays:

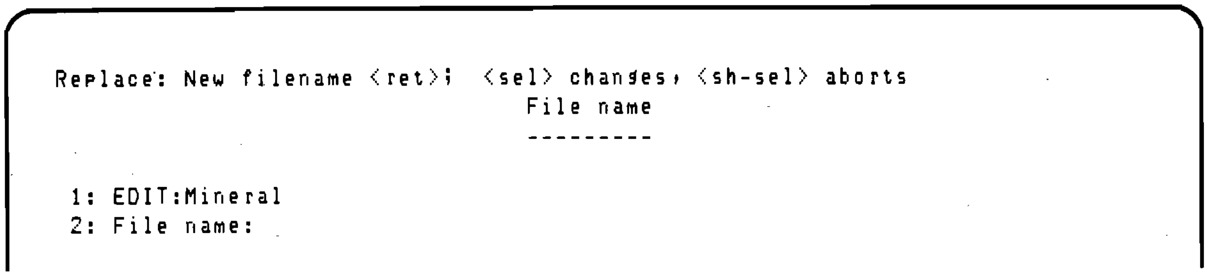
```
Insert: Filename <ret>;  empty line or <sel> leaves
                                  File name
                                  ---------

 1: EDIT:Mineral
 2  File name:
```

File number two was erased (as well as all file names below it), and the prompt `2 File name:` was printed, indicating that you are defining a new file number two. The old file number two has been bumped down to file number three.

5. Type `NewSecond` Return. Another prompt is displayed, asking for another file name. Press Return; we only wanted the one new file. Now your screen looks like this:

```
1: EDIT:Mineral
2: NewSecond
3: EDIT:Mineral@
4: ../Desk/LastFile
```

6. Since you added `NewSecond`, you've decided that the last two files are not really needed. Move the cursor to `3:` and press D for Delete. File three goes away, and the file below it moves up one position.
7. Press D again, and the new file three goes away.
8. Now let's modify file number two: `NewSecond`. The name should really be `EDIT:SecondOne`. Move the cursor to file number two, and press R. The screen displays this:



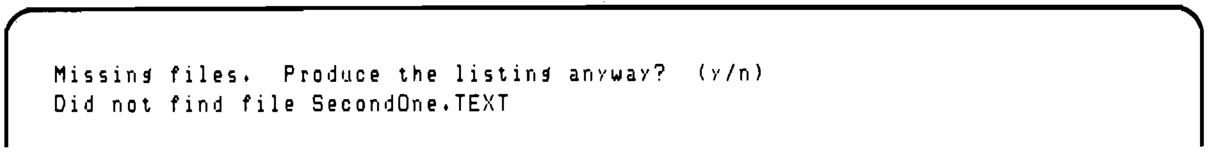
```
Replace: New filename <ret>;  <sel> changes, <sh-sel> aborts
                              File name
                              ---------

 1: EDIT:Mineral
 2: File name:
```

The old file name is erased, and the `File name:` prompt comes up, waiting for you to type a file name. If you changed your mind at this point, you could just press Return, and the old file name would come back. But we know we want to change the file name, so type `EDIT:SecondOne` Return in response to the `File name:` prompt.

Now the file pool looks like this:

```
1: EDIT:Mineral
2: EDIT:SecondOne
```

9. Press L. The list environment is again displayed.
10. Press L again. The Lister appends `.TEXT` to each file name in your list, then checks all of the files to see if they exist. It will not be able to find the second file, `EDIT:SecondOne.TEXT`. You will get the following message:



```
Missing files.  Produce the listing anyway?  (y/n)
Did not find file SecondOne.TEXT
```

11. Press Y. This time only the file `EDIT:Mineral.TEXT` will be printed. Nothing will be printed for the missing file.

12. Press the space bar to exit the List menu.
13. There is one more command the Lister has: Zap. As in the Editor, the Zap command in the Lister deletes things. In the Lister, Zap deletes the file names in your file pool – *all* of the file names. Press `Z`, and you will get the following warning:

```
WARNING: Want to zap 2 files? (y/n)
```

14. Press `Y`. All file names are deleted, but none of the listing parameters are changed. If you had pressed `N`, you would have been returned to the main Lister prompt, and nothing would have been changed. As you can see, this command is very similar to the Editor's Zap command.
15. Press `Q`. Your screen now displays:

```
Quit:
    Exit without updating
    Return to the lister without updating
    Write to a file name and return
    Archive file, write new file EDIT:MyDoc.Lst
    Save as file new file EDIT:MyDoc.Lst
    Overwrite as file EDIT:MyDoc.Lst
```

16. Press `E`. There is no need to save this list file. The Lister knows that you have changed the file since you started working with it, so it displays the following:

```
Are you sure you want to exit without updating?
     Type Yes  to Exit Without Update
     Type No   to Return to Lister
```

17. Press `Y` and you are out of the Lister.

Now you know how to use the Editor to create documents, and how to use the Lister to list multiple files as a single document. The Picture Processor program allows you to convert plot files into picture files that the Editor and Lister can use in illustrating your documents.

Picture Processor Tutorial

You can get a list of the graphics editors that work with HP TechWriter from your local Hewlett-Packard Sales Representative. Appendix C., *Making Your Own Pictures*, discusses some of the options available to you.

The plot files generated by these methods must be translated through the HP TechWriter Picture Processor. A step-by-step example of how to use the Picture Processor follows, and there is more information on the Picture Processor in the *Picture Processor Reference* chapter of this manual.

Running the Picture Processor Program

We will use the Picture Processor in this example to translate an unprocessed example file, Mohs, into a file you can use from the Editor or the Lister for document illustration.

1. Load the `LIST:` disc into a disc drive and shut the disc drive door.
2. Enter the Picture Processor by pressing P from the main command prompt of the HP TechWriter Environment, or by executing X `LIST:PICPROC.` from the Pascal Operating System's main command line.

   When you have started the Picture Processor, it displays the following message on your screen:

```
HP TechWriter PICTURE PROCESSOR [Rev. 1.0]  1-Jul-84

Copyright 1983 Hewlett-Packard Company.
        All rights reserved.

Plot file name <ret> (Just <ret> to quit):
```

3. The Picture Processor is asking you for the name of an unprocessed plot file. Type `LIST:Mohs` Return. The Picture Processor will find the file you've entered, then display the following message:

   ```
   New file name <ret> (Just <ret> to quit):
   ```

4. The Picture Processor is asking you where you would like it to write the processed file. Type Type `LIST:Mohs.P` Return. The following message appears on your screen:

   ```
   Processing....
   ```

   The Picture Processor will process the file and write it to file `Mohs.P` on the `LIST:` disc. When the processing has been completed, the Picture Processor will display:

   ```
   Picture processing of file LIST:Mohs is complete.
   Processed file LIST:Mohs.P is available for use.

   Plot file name <return> (Just <return> to quit):
   ```

5. Type $\boxed{\textbf{Return}}$ to leave the Picture Processor. You can now enter the Editor and use a `^^D f=LIST:Mohs.p` drawing command to display the picture you have just processed in your document. Note that, unlike the Editor and the Lister, the Picture Processor reads and produces files with the exact names that you type.

This completes the introduction to the document-related programs in the HP TechWriter package. The next chapter, *More About Editing*, has more examples showing various ways you can use Editor commands.

More About Editing

Chapter 4

Introduction

This chapter builds on the example from the last chapter, but gets substantially more involved with the power of the HP TechWriter Editor commands. If you have not read through the *Getting Started* chapter, we encourage you to do so; you will be much better able to understand this one.

Entering the Editor With an Old File

1. Put the `EDIT:` disc into an on-line disc drive (if it isn't already) and press E to re-enter the Editor.
2. When you're entering the Editor, and want to edit a file you created before, you specify the file name in response to the first prompt the Editor gives you:

```
HP TechWriter EDITOR [Rev. 1.0]  1-Jul-84

Copyright 1983 Hewlett-Packard Company.
         All rights reserved.

No workfile found.
File? (<ret> for new file, <stop> exits)
:
```

3. Respond by typing:

   `EDIT:Mineral` Return

   The Editor reads the file, and the main Editor prompt appears, and you're ready for action.
4. If you get a message like this:

   ```
   File: EDIT:Mineral.TEXT
   Error: file not found. File?
   :
   ```

   the Editor is telling you it can't find the file name you specified. The problem may be any of several things:

   a. You misspelled the volume name or the file name.
   b. You did not put a colon (:) between the volume name and the file name.

   c. You left out the volume name (and maybe the colon, too), and the default volume – that volume where a file is sought unless you specify otherwise – is not `EDIT:`.
   d. The `EDIT:` disc is not on-line at the moment.
   e. The disc drive is not turned on.

   If you had a problem, correct it, and let's continue the tutorial.

5. The next section deals with formatting text. We will not be discussing pictures again until the section *Getting Drawings into Your File.* To simplify examples, the drawing command `^^D` and its terminator `^^` will usually not be shown when the file contents are shown. For this reason, go into the environment and disable picture-drawing on the screen:
   a. Press (S) (E). This enters the environment-setting mode.
   b. Press (D) (F). This sets `Drawing False`; the Editor will not draw any pictures again until told to do so.
   c. Press the space bar to exit the environment.

Before we go on, let's learn about the Help command.

Help! (The Command)

After you are familiar with the Editor, and you are using it without this manual close by, there may be occasions when you know what a command does, but you can't think of its name. Or, you may know the command and its name, but can't remember which option you need. For cases like these, the Help command can be useful.

1. Press H. The text on the screen disappears, and a screenful of "Help" text describing the regular Editor commands is displayed. Its prompt line looks like this:

```
>Help 1:  <arrow keys>  <sel> or <sp> leaves
```

2. Press ▶. This displays the next screenful of information which describes the document commands. Its prompt line looks like this:

```
>Help 2:  <arrow keys>  <sel> or <sp> leaves
```

3. Press ◀. The first screen of Help text reappears. By pressing ◀ and ▶, you can alternate between the two pages of Help text.
4. Press Select, and the Help information goes away, and your text comes back.

Now let's learn more about moving the cursor.

Moving the Cursor

We have moved the cursor around several times in the course of our example. Every time we moved it, it was because we needed to indicate to the Editor *where* to do a particular operation. In the previous tutorial, the arrow keys and the **Return** key moved the cursor. In this section, you'll be using other methods of moving the cursor. They will let you:

- Move the cursor faster,
- Move the cursor to specific locations in your file, and
- Move the cursor specific amounts from where it is currently.

With the text you have on the screen, experiment with positioning the cursor. The four arrow keys, the **Return** key, the **Tab** key, and the space bar all move the cursor. You should be familiar with these things from the *Getting Started* chapter.

Repeat Factors in Cursor Movement

A "repeat factor" is a number specified before a cursor-moving key or command which is a shortcut method of pressing the key or typing the command that many times. Any integer in the range 1 to 9999 can be used as a repeat factor before any of the cursor control keys and most of the Editor commands.

The result of typing a repeat factor before a cursor-moving key will be the same as if you had pressed the key that many times.

1. Use the arrow keys to move the cursor to the beginning of the file. Type the number 42 and then press the space bar. This causes the cursor to move 42 characters in the current direction.
2. The **/** key can also be used as a repeat factor; it means "as many as possible." Press **/** **▶**. This moves the cursor to the end of the file, and will work regardless of where the cursor is, and regardless of the length of the file.

Moving to the Beginning and End of Your File

1. The Jump command offers another means of cursor positioning. Press **J** and the top of your screen displays:

```
>JUMP: Begin End Marker  [<sh-sel> to leave]
```

   Typing **B** causes the cursor to jump to the beginning of the file; in the case of our example, to the "T" of "The" in the first line and the Editor's main prompt reappears. Do this, and notice that wherever the cursor was, it is now at the beginning of the file.
2. Now press **J** then **E** and the cursor moves to the end of your file. The beginning and end of a file are simply the first and last characters in the current text file.

Moving to Markers

You can also Jump to previously set locations, called *markers*, by typing J M followed by a marker name. After a marker is defined, you can jump to it from any place in the file. The *Set* command lets you define markers.

Markers are set by moving the cursor to where you want the marker, pressing S M (for Set Marker) and typing in a marker name followed by Return. The Editor accepts up to eight characters for a marker name. All commonly-used characters except blanks and commas are allowed in marker names. If commas are specified for a marker name, they are stripped out. The Editor converts these marker names to upper case so they can be typed in whatever form is convenient.

1. Let's set a marker in our file now. Move the cursor to the "M" in "Mohs". Press S M, and you are prompted for a marker name:

```
Set what marker?
```

2. Type `Mohs` Return. The main Editor prompt comes back, and nothing visible has happened. However, a marker named `MOHS` now exists in your file.
3. Now you can go directly to this point from wherever else in the file you happen to be. Jump to the end of the file with J E.
4. Now press J M to tell Editor you want to jump to some marker. The Editor responds:

```
Jump to what marker?
```

5. Type `mohs` Return (again, letter case is insignificant in marker names). The cursor will immediately move to the first letter in `Mohs`.
6. If you tell the Editor to go to a non-existent marker, you will get an error. Try pressing J M Z Return. This tells the Editor to jump to marker "Z". There *is* no marker Z, so the Editor beeps and responds:

```
>ERROR: Not there. <space> continues.
```

Press the space bar to clear the error condition.

Up to ten markers can be set in a file at one time. If you attempt to set more than ten, the Editor displays the markers in a numbered list and prompts you for the number of the marker you wish to replace. Specify the marker to be replaced by typing its number (0 through 9), not its name.

Do not do this now, but if you wish to delete *all* the currently defined markers, you would use the *Zap markers* command in environment-setting mode. There is no way to delete markers one at a time.

Markers are also used with the Copy command – you can copy *pieces* of disc files into your text (more about this later). Marker names are shown in the environment display. Use marker names that mean something to you, since the location in the text is not shown once they are set.

Moving a Screenful at a Time

The Page command lets you move through a file one screen (23 lines) at a time. It is roughly equivalent to using a repeat factor of 23 with ▲ or ▼ depending on the direction shown in the prompt. You can move *n* screens distance by pressing *n* P, where *n* is a repeat factor between 0 and 9999.

1. Jump to the beginning of the file by pressing J B.
2. Press P. The next screenful of text is printed, and the cursor remains in the same place, relative to the screen.
3. Press < to set the current direction backward, and press P again. The previous screenful of text is again displayed.
4. Press > to set the current cursor direction forward so we can continue the example.

Finding What You Typed Before

Perhaps the most powerful cursor-moving command is Find. It allows you to look for almost anything that you've placed in your text. Let's find some things in our text. The find command finds *strings*, which are merely sequences of characters. Strings may be any length between zero and 128 characters. The Editor starts searching at the cursor position, and proceeds in the direction indicated by the cursor-direction indicator (that little ">" or "<" at the beginning of the prompt line).

As of now, your paragraph should look like this:

```
>TWedit:  Adjust  Copy  Delete  Find  Insert  Jump  Replace  Quit  Xchange  ?
    The German mineralogist Friedrich Mohs (1773-1839) developed a scale of
 hardness in order to assign a definite figure to the values he obtained in
 determining mineral hardness. He used the scratch-hardness method,
 observing the comparative ease or difficulty with which one mineral is
 scratched by another.
```

1. Move the cursor to the beginning of the file by pressing J B.
2. Press F. This invokes the Find command, and the following prompt is displayed on the screen:

```
>Find[1]: T <target>=>
```

3. Type the following:

   `/to/`

   The Editor uses characters called *delimiters*, which delimit the endpoints of a string; that is, they tell the computer where the string of character starts and stops. In this case, you typed a slash as the first delimiter ("the string starts here"), then the target string (the thing you're looking for), and another of the same delimiter ("the string ends here").

   Now the cursor will move to the space after the word "to" in the phrase "to assign" in the second line of the paragraph. This was the first occurrence of "`to`" the Editor found.
4. The delimiters (slashes in this example) are indispensable. The Editor doesn't know where strings start and stop otherwise. To illustrate this, press F and type the following:

   `order`

   Immediately when you pressed the "`o`" in "`order`", the machine beeped and stated:

```
>ERROR: Invalid delimiter. <space> continues.
```

and all the rest of the characters were ignored. What the computer is saying is that the first character typed wasn't a legal delimiter character. The Editor must have delimiters so it knows the exact beginning and ending characters of the string. Legal delimiters are punctuation characters and most other special characters, such as /, #, $, and @. Blanks are not legal delimiters but may be used in a string. The complete list of legal delimiters is in the *Editor Reference* chapter.

Delimiters are used in pairs – you must use the same delimiter character to end the string as you did to start it.

The message:

```
>ERROR: Invalid delimiter. <space> continues.
```

is still there. The "<space> continues." tells you to press the space bar to continue with the program. Do so now, and the main Editor prompt comes back.

5. Jump to the beginning of the file. Find the next occurrence of "order": type F /order/. Now the cursor is in the space after "order" in the phrase "in order to" in the second line of the paragraph.
6. Finally, find the next occurrence of "order": type F /order/. This time, the machine beeps, and this prompt is displayed at the top of the screen:

```
>ERROR: Pattern not found. <space> continues.
```

7. There weren't any more occurrences of the string "order" in the file, so the Editor is notifying you. Again, where the prompt says <space> continues, it is telling you to press the space bar to go on. Do so.

Finding Exact Strings of Characters

What we were finding with the Find command in the previous section were *literal strings.* A *literal* string is exactly that; a string used exactly as typed. No assumptions are made about what is surrounding it.

There is another kind of thing you will probably want to look for, and we'll talk about it after we see one of the pitfalls of finding *exactly* what you specified.

Suppose that the hardness scale described in the paragraph had been developed by Mohs' (fictitious) daughter, instead of himself. All the male pronouns in the text ("he") would need to be changed to "she" instead. The Find statement allows us to locate the occurrences of "he" that need to be changed.

1. Go to the beginning of the file by pressing J B.

2. Type:

   F `/he/`

   Already we have a problem! The Find statement caused the Editor to find the "he" embedded inside of the word "The" at the beginning of the paragraph. Obviously, we shouldn't change this "he" to a "she."
3. Type F `/he/` again. It now finds the "he" in "the". We could type F `/he/` yet again, but wouldn't it be easier to tell the Editor to look for the same thing it looked for last time? We can do this.
4. Press F S. The S means "the *S*ame string you looked for last time." This is a shortcut method of typing `/he/` over and over again. Keep pressing F S until the message:

```
>ERROR: Pattern not found. <space> continues.
```

   is displayed. The Editor will have found these occurrences:

   "he" in "The"
   "he" in "the"
   "he" in "he"
   "he" in "the"
   "he" in "the"
   "he" in "scratched"
   "he" in "another"
5. Press the space bar to clear the error.

There are occurrences of "he" all over, but only one is the kind we're looking for! The ones embedded in other words should not be changed, but how can the Editor tell the difference?

Finding Words

There is a "switch" which tells the Editor to look for either "literals" or "tokens" during searches. We saw above what a literal is; it is a string sought for exactly as typed. On the other hand, a "token" is a string with one assumption made: *it will be surrounded by non-alphanumeric characters.* A space is a non-alphanumeric character; therefore, if we did token searches for occurrences of "he", it would find only the *words* "he" and not the *literals* "he". This is exactly what we need.

There are several kinds of non-alphanumeric characters which mark the beginning or end of a token:

- Any punctuation character: period, comma, colon, apostrophe, space, etc.,
- Any arithmetic operator: plus sign, minus sign, asterisk, parentheses, etc.,
- Any "special" character: Pound sign, backslash, grave accent etc.,
- The beginning or end of a line, and
- The beginning or end of a file.

Let's tell the Editor to find only tokens.

1. Press J B to get to the beginning of the file.
2. Press F to invoke the Find command. The following prompt comes up:

```
>Find[1]: T <target>=>
```

   Notice the "T" in the prompt. This indicates that you can press T before you type the target string. The "T" causes the next search to be a search for tokens, not literals. "T" has a special meaning, then, and is not part of the target string, nor is it a delimiter.

3. Press T. Nothing visible happens, except that a "t" is printed at the top of the screen, right after the prompt. But now the Editor knows that you want a token search.
4. Type `/he/`. The Editor skips the occurrences of "he" which occur in "The" and "the" because those occurrences are not tokens. The first token "he" the Editor finds is in the phrase "he obtained".
5. Press F T S. The Editor responds:

```
>ERROR: Pattern not found. <space> continues.
```

   There are no more occurrences of the token "he" in the text. Press the space bar to continue.

If you are searching through a large file for a token, it might become annoying, having to press T before the first delimiter of every Find command; the T is easily forgotten.

The literal search is the default kind of search, and you can tell the Editor to use a token search *for this Find command.* We've already seen that. Now we'll learn how to tell the Editor to use a *token* search by default unless we specify a literal search for a specific case.

1. Press S E. This tells the Editor that you want to *S*et the *E*nvironment. A display similar to the following comes up:

```
>Environment: {options} <sel> or <sp> leaves

  Auto indent   False                       Command ch    ^
  Filling       True                        Underline ch  \
  Justify       False                       Escape ch
  Left margin   1                           Draw figures  False
  Right margin  75                          Token search  False
  Para margin   5                           Ignore case   False
  Zap markers

  394 bytes used, 203774 available.
  Patterns:
    <target>= 'he'
  Markers:
    MOHS
  File EDIT:Mineral.TEXT
  Created: 10-6-84  Used: 10-6-84  Type: TEXT
```

This is a display of all the switches you may set. There is a lot more here than we need to deal with at the moment.

2. For now, press T. This tells the Editor that you want to enter a value for the parameter starting with "T": `Token search`.
3. Press T again. This sets the value to `True`. Press the space bar to exit the environment-setting menu, and your text comes back.
4. Go back to the beginning of your file by pressing J B.
5. Now press F. The following prompt appears at the top of your screen:

```
>Find[1]: L <target>=>
```

Notice that the "`T`" in the prompt has been replaced by an "`L`"! (Do not press L now.) The Editor is allowing you to choose a literal search, now that you've set the default search to Token. In short, whether the default search type is literal or token, the other type is always available for individual searches.

6. Now type `/he/`. Notice that the Editor now ignores all occurrences of "he" which are embedded inside other words.

7. Press (F) (S) again. This time, the message comes up:

```
>ERROR: Pattern not found. <space> continues.
```

indicating that the Editor can't find any more of the *token* "he". It only finds the kind we want – the *word* "he" – and our question is answered: there is only one occurrence of "he" that we would have to change to "she".

Or is there?

8. Before we go on, we must respond to the error message. Press the space bar to do so.

Ignoring Upper and Lower Case

You have probably already noticed that although there is only one occurrence of "he" that needs to be changed, there is an occurrence of "He" which also needs to be changed. The first word of the second sentence is "He", but, of course, since it is the first word in the sentence, it is capitalized. This illustrates an important point in the HP TechWriter Editor: **a capital letter *in the text* is different from a lowercase letter**.

This explains why, when we were looking for "he", we didn't find "He". The string "he" is different from the string "He" because the letter "h" is different from the letter "H".

So how do we tell the Editor to include both kinds of words: those with capital letters and those without? There is another internal switch the Editor uses during searches. This switch determines whether the Editor ignores upper or lower case.

1. Press (S) (E). Again, this tells the Editor that you want to *S*et the *E*nvironment. The following display comes up:

```
>Environment: {options} <sel> or <sp> leaves

  Auto indent   False                       Command ch     ^
  Filling       True                        Underline ch   \
  Justify       False                       Escape ch
  Left margin   1                           Draw figures   False
  Right margin  75                          Token search   True
  Para margin   5                           Ignore case    False
  Zap markers

  394 bytes used, 203774 available.
  Patterns:
    <target>= 'he'
  Markers:
    MOHS
  File EDIT:Mineral.TEXT
  Created: 10-6-84  Used: 10-6-84  Type: TEXT
```

2. Press $\boxed{\text{I}}$. The cursor moves to the `Ignore case` parameter and waits for the parameter value: `True` or `False`.
3. Press $\boxed{\text{T}}$. The `Ignore case` switch now has a value of `True`, meaning the case of letters will be ignored during searches.
4. Press the space bar to leave the environment-setting menu.
5. Move to the beginning of the file with a Jump to the Beginning.
6. Type $\boxed{\text{F}}$ $\boxed{\text{S}}$. The Editor places the cursor after "he". So far, so good.
7. Type $\boxed{\text{F}}$ $\boxed{\text{S}}$ again. This time, the Editor finds "He". It is successfully ignoring letter case while searching for strings.

We have now covered the most important ways of moving the cursor from where you are to where you want to be. This knowledge is useful when you want to create text, because creating text takes place at the cursor position.

Creating Text

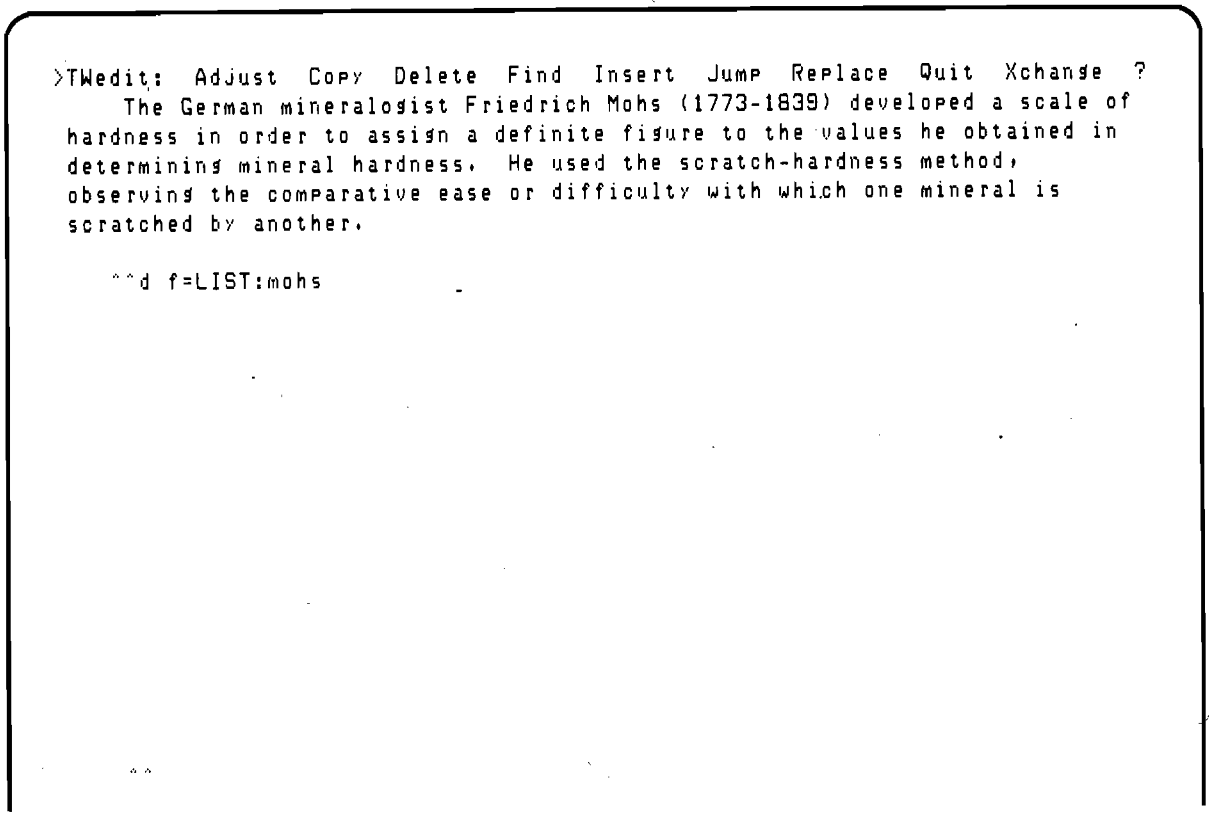
In the last section, we inserted a small portion of text into our file: " `in order`". Inserting larger portions of text works similarly; pressing (Return) while in Insert mode causes subsequent text to be placed on the next line. The text that already exists before Insert was invoked is temporarily erased from the screen at the time the cursor advances a line. It comes back when you either accept the insertion with (Select) or abort the insertion with (Shift)-(Select).

Note

The Insert command margins the text when in Document mode. Any time you do an Insert and confirm the operation by pressing (Select), all paragraphs inserted are automatically margined, unless a local margin command forbids it (more about this later). However, if you do an Insert and abort the operation by pressing (Shift)-(Select), no automatic margining is done.

When entering several paragraphs while in document mode, the Editor keeps track of where the insertion started. It marks this location by setting an "anchor" at the cursor position at the time the Insert command was invoked. When you get done with the large insertion and press (Select), the Editor will margin all paragraphs between the anchor (the cursor position at the start of the insert) and the cursor position at the end of the insert. There will be more discussion of the anchor later.

Your text should look like this right now:



```
>TWedit:  Adjust  Copy  Delete  Find  Insert  Jump  Replace  Quit  Xchange  ?
     The German mineralogist Friedrich Mohs (1773-1839) developed a scale of
 hardness in order to assign a definite figure to the values he obtained in
 determining mineral hardness.  He used the scratch-hardness method,
 observing the comparative ease or difficulty with which one mineral is
 scratched by another.

    ^^d f=LIST:mohs
```

Let's add some more text to this report.

1. Press $\boxed{\text{J}}$ $\boxed{\text{B}}$. This moves the cursor to the beginning of the file.
2. Type $\boxed{\text{F}}$ `/^^d/`. Notice that this Find command will work regardless of whether the drawing command's first character is "d" or "D"; `Ignore case` is still `True`.
3. Move the cursor to the beginning of this line and press $\boxed{\text{I}}$. You just invoked the Insert command.
4. Type the following:

```
     Another scale of hardness, called the Vickers hardness scale, was
created during World War II.  In the Vickers method of determining
hardness, a pyramidal diamond is pushed into the material being tested for
about fifteen seconds, under a specified load.  The indenter is removed,
and the surface area of the indentation is measured under a microscope.

     A graphical comparison of Mohs and Vickers hardness numbers is shown
below.
```

5. Press $\boxed{\text{Return}}$ $\boxed{\text{Return}}$. Next, we'll press $\boxed{\text{Select}}$, *but before you do*, be prepared to look at the top line of your screen. When you're ready, press $\boxed{\text{Select}}$.
6. In case the message flashed by too quickly for you to read, here is what it said:

```
>Margin[i]..
```

This message was indicating that a multi-paragraph margining operation was being done. All the text between the anchor and the final cursor position was margined when you pressed $\boxed{\text{Select}}$.

If you insert a great deal, the margining process at the end may take several seconds. To let you know that there is something going on, periods will be printed. In general, each period indicates that another paragraph was just margined.

If you never make misteaks, you may skip over the following section, which deals with the correction of errors in the text.

Altering Text

Mistakes or necessary changes in a text file or program are not always obvious when creating the text. The Editor features commands which allow you to go back and make changes when needed. These are the Replace and eXchange commands and the Delete/Insert sequence. These were covered minimally in the *Getting Started* chapter, but they will be covered more extensively now. They will be demonstrated by making alterations to the sample text, which should look like this now (excluding the drawing):

```
>TWedit:  Adjust  Copy  Delete  Find  Insert  Jump  Replace  Quit  Xchange  ?
    The German mineralogist Friedrich Mohs (1773-1839) developed a scale of
hardness in order to assign a definite figure to the values he obtained in
determining mineral hardness.  He used the scratch-hardness method,
observing the comparative ease or difficulty with which one mineral is
scratched by another.

    Another scale of hardness, called the Vickers hardness scale, was
created during World War II.  In the Vickers method of determining
hardness, a pyramidal diamond is pushed into the material being tested for
about fifteen seconds, under a specified load.  The indenter is removed,
and the surface area of the indentation is measured under a microscope.

    A graphical comparison of Mohs and Vickers hardness numbers is shown
below.
```

One-for-One Character Replacement

One way to modify text is by using the eXchange command. Say, for example, that we want to change "determining" in the first paragraph to "finding".

1. Move the cursor to the "d" in "determining" (*first* paragraph, remember).
2. Press X. This puts the Editor into exchange mode, and the prompt at the top of the screen looks like this:

```
>Xchange: <text> <bs>  [<sel> changes, <sh-sel> aborts]
```

   Characters will be replaced one-for-one until Select or Shift-Select is pressed.

3. Type `finding++++`. Obviously, we don't want the ++++ to be in the file, so press Shift-Select. You'll notice that all of the original text comes back, and the main Editor prompt reappears.
4. Now press X again and type the same thing: `finding++++`. We still don't want the ++++ in there, so press Backspace four times, and watch the screen. You'll notice that the original text reappears, one character at a time. This shows that if you make a typing error, a simple backspace will suffice to void the error; there is no need to press Shift-Select and start completely over.

5. Type four blanks where the four plus signs were, and press (Select) to accept the change. Now you have the correct text, but there is a huge hole in the line, and it looks awful.
6. Press (M), and the paragraph is remargined.
7. In the same way you did in the section *A Sample Editor Session*, replace the "`finding`" with "`determining`", so we can continue. Move the cursor to the beginning of the line, and type:

   (R)`/finding//determining/`
8. Since "determining" is a longer word than "finding," the paragraph needs to be remargined. Do so.

When exchanging characters, there may be situations where you want to exchange some characters near the left end of the line, and some more characters near the right end of the line. One nice feature of the eXchange command is that you can move the cursor to the right *while in eXchange mode.*

1. Move the cursor to the "G" of "German mineralogist".
2. Press (X) to invoke the eXchange command.
3. Type a few characters, observing the exchanging as you go.
4. Move the cursor several characters to the right by pressing (▶).
5. Type some more characters, observing the exchanging as you go.
6. Press the space bar, note that spaces are entered.
7. Press (Shift)-(Select) to abort the experiment.

Notice that you don't have to exit the eXchange command, move the cursor, and re-enter the eXchange command. You should always position the cursor before initiating eXchange because this command cannot cross line boundaries; you can only make eXchanges on the line where the cursor is located, and only to the right of the cursor position at the point where eXchange was invoked.

Try this:

1. Move the cursor to the first line of the first paragraph.
2. Press (X), invoking the eXchange command. Type some characters. This all works fine.
3. Press (Return). The computer merely beeps; you cannot change lines from within an eXchange.
4. Abort the experiment by pressing (Shift)-(Select).

The eXchange command is handy, but when you need to change many occurrences of one string to another string, it would become tedious to move the cursor, invoke eXchange, type the new string, and press (Select) for every occurrence. For this kind of operation, the Replace command fills the bill.

Replace

The Replace command allows you to replace occurrences of one string with another string. The old string need not be the same length as the new string. When a string is replaced by another, margining is *not* automatically done, as it is during an Insert. Searching for strings to replace is done in the current direction.

The Replace command is similar to the Find command in many ways. They use delimiters the same way, `Token search` and `Ignore case` have the same definitions for both commands, and are used by both commands. The "`T`" or "`L`" in a Replace prompt does exactly the same thing as the "`T`" or "`L`" in a Find prompt.

There are two things you should remember from the discussion of repeat factors:

- Putting a number before a cursor-moving key means the same thing as pressing that key the indicated number of times, and
- Putting a slash before a cursor-moving key means "press the key as many times as possible."

Repeat factors can be used in the Replace command, too. Typing a number before invoking the Replace command tells the Editor to make the indicated number of replacements. The slash tells the Editor to attempt to replace *all* subsequent occurrences of something with something else.

Let's do the modification we were considering earlier, during the Find discussion.

1. Jump to the Beginning of your file.
2. Set the environmental parameter `Ignore case` back to `False`: S E I F. (The section *Ignoring Upper and Lower Case*, several pages back talked about this.) Press the space bar to return to your text.
3. Press 9 9 9 9 R. The following prompt is shown:

```
>Repl[9999]: L V <target><repl>=>
```

4. Type:

   ```
   /he//she/
   ```

   The Editor beeps, and displays:

```
>WARNING: Replaced 1, <space> continues.
```

   When you use an explicit repeat factor which is larger than the number of occurrences in the rest of the file, the Editor tells you how many *were* replaced. It is often a good idea, when replacing a large number of occurrences of a string, to use a specific number for the repeat factor (for example, 9999) rather than the infinite repeat factor ("/"). This way, the Editor will tell you the number of occurrences replaced, and if the number is unreasonable, you will know immediately.

5. Press the space bar to clear the warning message.

Verifying Replacements

The other letter in the Replace prompt, "V", specifies that you want Verify mode on. With Verify on, every tentative replacement is first checked out with you, and you must specify either:

- Yes, replace this occurrence. You specify this by pressing R.
- No, do not replace this occurrence. You do this by pressing the space bar.

Any other character pressed will be ignored; you must press R or the space bar. To leave the entire command, you may also press Shift-Select.

Let's use the Verify option of the Replace command. We realized, after we typed the last paragraph of our example text, that the word "Mohs" is possessive, and thus needs an apostrophe, indicating *his* numbers.

1. Jump to the Beginning of the file.
2. Invoke the Replace command with the slash ("as many as possible") for a repeat factor (/ R) and type:

   ```
   v/Mohs//Mohs'/
   ```

   After you type the final delimiter, the cursor is positioned immediately to the right of the first occurrence of the target string. This is in the first sentence of the first paragraph. The Editor then displays:

```
>Repl[/]: R replaces, ' ' doesn't, <sh-sel> aborts
```

3. Press the space bar; we do not want this occurrence to be replaced.

   Now the cursor is placed after the "Mohs" in the last paragraph.
4. This is the one we want to replace, so press R to replace this occurrence.

If we had only wanted to verify the replacement of the next five occurrences, we could have pressed 5 R V. The prompt line would then have shown:

```
>Repl[5]: L V <target><repl>=>V
```

There is one thing to keep in mind when replacing a finite number of occurrences of a target string *with verification activated*: there will be the specified number of *tentative* replacements, not *actual* replacements. For example, if you specify a repeat count of five, with verification on, you will be asked to confirm or deny five occurrences, regardless of whether you authorize the replacement. It will **not** keep asking until you have confirmed (by pressing R) five replacements.

You can try this very simply:

1. Jump to the Beginning of the file.
2. Press `3` `1` `R` `L` `V` `/he//she/`. The Editor finds the first *literal* occurrence and asks whether you want to replace it or not. Remember, "literal" includes those occurrences embedded within other words.
3. Press the space bar to say "No, don't replace this one" for all occurrences of "he" except for that one embedded in "she". For that one, press `R` to say, "Yes, replace that one."

   There were fourteen occurrences of "he" you said "no" to, and one occurrence you said "yes" to, for a total of fifteen occurrences of "he" in the whole file. The following warning appears after fifteen verifications:

```
>WARNING: Verified 15, <space> continues.
```

   The Editor couldn't find the 31 replacements you specified. The repeat factor is the number of confirmations plus the number of denials, *not* merely the number of confirmations.
4. Press the space bar to clear the warning message.
5. Jump to the Beginning of the file, and Replace the "sshe" we just created with "he".

Your text should once again look like this:

```
>TWedit: Adjust Copy Delete Find Insert Jump Replace Quit Xchange ?
     The German mineralogist Friedrich Mohs (1773-1839) developed a scale of
  hardness in order to assign a definite figure to the values he obtained in
  determining mineral hardness. He used the scratch-hardness method,
  observing the comparative ease or difficulty with which one mineral is
  scratched by another.

     Another scale of hardness, called the Vickers hardness scale, was
  created during World War II.  In the Vickers method of determining
  hardness, a pyramidal diamond is pushed into the material being tested for
  about fifteen seconds, under a specified load.  The indenter is removed,
  and the surface area of the indentation is measured under a microscope.

     A graphical comparison of Mohs and Vickers hardness numbers is shown
  below.
```

To Verify or Not to Verify?

If you are *sure* you want the specified number of occurrences to be replaced, don't press V before specifying the strings. Then no confirmation will be necessary before each replacement. This can be somewhat hazardous, especially when using Literal mode, as the following example shows.

Assume that the direction is forward (>) with the cursor located at the beginning of the file, and you want to replace all occurrences of the word `he` with `she`, as discussed previously. A literal replacement would substitute `she`'s for the `he`'s embedded in the words `The`, `the`, `scratched`, and `another`, resulting in `Tshe`, `tshe`, `scratcshed`, and `anotsher`. If you did a replacement without Verify, you wouldn't notice the incorrect replacement until you happened to stumble upon it.

Re-Using the Same Strings

A shortcut which is available for both the Find and Replace commands is the "same" option. We've already seen how this worked with Find, now let's see how it works with Replace.

The target string, the replacement string, or both can be replaced by the pressing S. This tells the Editor to use the same target or replacement string it used last time. Which old string is used is determined from the position of the "S" in the input.

Let's use the "same" option in our file.

Suppose you wanted to replace all occurrences of "hard" with "soft", with verification.

1. Go to the beginning of the file.
2. Type / R `v/hard//soft/`. The Editor, being in token mode, finds no occurrences of "hard". The screen displays:

```
>ERROR: Pattern not found. <space> continues.
```

3. Press the space bar to clear the error message.
4. In order to find "hard", you must do a literal search. Type / R `lvss`. This means "With an infinite repeat factor, replace the literals with verification, using the same target and replacement strings as last time."

   Now, in literal mode, the replace-with-verify operation takes place as you wanted it to in the first place. Using the same strings as last time can be a nice time saver.
5. We have already demonstrated what we wanted to demonstrate, so press Shift-Select to abort this replace operation.

If you have not specified a pattern for Replace (or Find), and you try to use the "same" option, the computer will beep and give you the error message:

```
>ERROR: No old pattern. <space> continues.
```

Ignoring Upper and Lower Case

There is one more thing about Replace. In the environment, the parameter `Ignore case` affects Replace operation like it does the Find operation. However, the Replace command requires two strings – the target and the replacement – while the Find only requires the target. The effect of `Ignore case` on the targets is the same for both commands. The replacement string, though, is not affected by `Ignore case`; *it is inserted into the text exactly as typed.* There is no uppercase or lowercase conversion done on the replacement string.

If `Ignore case` is set to `False`, the searching for the target will only find exact matches, just as Find does. If `Ignore case` is `True`, strings will be found whether the string has upper case characters or lower case characters in it. Let's use the replace command to show us what it does when ignoring case.

1. Set `Ignore case True` in the environment.
2. Jump to the Beginning of the file.
3. Type:

   `/` `R` `T` `V` `/he//they/`
4. Press `R` for all tokens found, and notice that not only was "he" replaced with "they", but "He" (capitalized), at the beginning of the second sentence of the first paragraph was also replaced with "they". The second sentence now starts with "they used the...".
5. Using the eXchange command, change "they" back to "he" and "He".

Similarities Between Find and Replace

There are several areas where the Find and the Replace commands are closely related or interact with each other. We have covered them separately, but here is a summary of the them. They are:

- The `Token search` parameter in the environment affects searches identically.
- The `Ignore case` parameter in the environment affects searches identically.
- When using the "same" option, both Find and Replace use the same target string. Thus, when using the "same" option from Find, you will get the target string used in the last Replace, if that was used since the last Find. The reverse is also true.
- Repeat factors can be used for both Find and Replace. If a finite repeat factor specifying a number larger than the number found is used, the Editor tells you how many were found.
- Both Find and Replace take place from the current cursor position and proceed in the current cursor direction, which is indicated by the first character of the command line.

Next, let's store the file as it stands now and then continue with the editing.

Storing a Previously-Stored File

1. Press Q. This time, the Editor knows what the file name will probably be, because you named it when you entered the Editor. Here is what the option menu looks like:

```
>Quit:
     Update the workfile and leave
     Exit without updating
     Return to the editor without updating
     Write to a file name and return
     Archive file, write new file EDIT:Mineral.TEXT
     Save as file new file EDIT:Mineral.TEXT
     Overwrite as file EDIT:Mineral.TEXT
```

   There are several options available from the quit option.
2. Press S; this stores a new copy of the file under the same name as before. Unlike the `Archive` option we used earlier, the `Save` option does *not* keep a copy of the existing file under a new file name.
3. Now press R (for the Return option) and you will be returned to the main Editor Prompt.

All the Quit options shown are explained in detail in the "Editor Reference" chapter. Now, let's learn how to remove text already in your file.

Removing and Recovering Text

The Delete command was introduced in *A Sample Editor Session* in the *Getting Started* chapter. We'll learn more about how to use it now.

1. Position the cursor somewhere close to the middle of the second paragraph and press D. This initiates the Delete command. The prompt line will look like this:

```
>Delete: < > <arrow keys>      [<sel> deletes, <sh-sel> aborts]
```

2. Press ▶ to move the cursor to the right. Characters disappear as the cursor moves.
3. Press ◀ to move the cursor back over the area where text has been removed. Notice that the text comes back. As you pass the point at which you initiated the Delete command, text is deleted toward the beginning of the file. Thus, you can remove text in either direction.
4. Press ▲. Observe how text disappears.
5. Press Return until text stops vanishing. You will be just before the ^^ at the end of the file.
6. The cursor direction can be redefined while in the Delete command. Press < to set the cursor-direction backwards.
7. Now that the cursor direction is backwards, press the space bar a few times and observe how the text is restored toward the beginning of the file.
8. Now press > to restore the cursor direction to forward, and press Shift-Select to abort this experiment.

Let's modify the first paragraph.

1. Move the cursor to somewhere in the first paragraph and press M to margin the paragraph. (This is merely to insure that your paragraph initially looks like the manual's.) It should look like this:

```
     The German mineralogist Friedrich Mohs (1773-1839) developed a scale of
hardness in order to assign a definite figure to the values he obtained in
determining mineral hardness. He used the scratch-hardness method,
observing the comparative ease or difficulty with which one mineral is
scratched by another.
```

2. Place the cursor under the "s" in the word "scale" in the first line, and then press D for Delete.
3. Type 3 followed by Return Select. This uses a repeat factor and moves the cursor three lines, deleting text as it goes. Your paragraph should now look like this:

```
     The German mineralogist Friedrich Mohs (1773-1839) developed a observing !
scratched by another.
```

   The text has been deleted, but it has been placed in a temporary storage area called the *copy buffer.*

4. Press C B, for Copy *B*uffer, and the contents of the copy buffer will be inserted at the cursor position, returning the text to its original state.

5. Press (C) (B) again, and another copy of the deleted text appears. This, then, is the way to duplicate blocks of text. You can move a block of text by deleting it, moving the cursor to the desired position, and retrieving the copy buffer. Cursor movement does not affect the contents of the copy buffer, but many other commands do, so if you want to retrieve the contents of the copy buffer, *do it as soon as possible* to get the text back into your file for safekeeping.
6. Now delete *both* copies of the formerly deleted text:
   a. The cursor should already be at the "s" in "scale". If it is not, move it there.
   b. Press (D) (6) (Return) (Select). Your text should again look like this:

```
          The German mineralogist Friedrich Mohs (1773-1839) developed a observing !
     scratched by another.
```

7. Note that there is some text extending off the right edge of the screen. This is indicated by an exclamation point (!) on the right-hand side of the screen.

   To make the off-screen text visible again, press (M) to margin the paragraph. It should then look like:

```
          The German mineralogist Friedrich Mohs (1773-1839) developed a
     observing the comparative ease or difficulty with which one mineral is
     scratched by another.
```

8. Now press (C) (B), and you'll get the message:

```
>ERROR: Invalid copy. <space> continues.
```

   Margining destroys the contents of the copy buffer.
9. Press the space bar to clear the error condition.

Take care not to wait too long or depend on the copy buffer too heavily as there are other commands (such as (M)) which alter the contents of the buffer. None of the cursor movements alter the buffer in any way.

Fast Removal of Text

Another method for removing text is the Zap command. It deletes all text between the cursor position and the **anchor point**. The anchor point is a position in the file which is set by the commands Find, Adjust, Insert, and Replace (remember their initial letters: "FAIR"). Upon entering the Editor, the anchor is at the first character in the file.

Before doing a Zap of some text, it is wise to confirm the position of the anchor by pressing (=). The equals key places the cursor at the current anchor point. (This is the final method of moving the cursor.)

1. Go to the beginning of your file.
2. Type (F) `/with/`. This places the cursor in the space following the word "with", and the anchor at the "w" of "with".
3. Press (=). The cursor moves to the "w" in "with".

If the anchor is not at the correct position, you can put it there by using either the Insert command with an aborted entry, or the Adjust command, moving the line zero spaces. You can set the anchor point to the cursor position, then, by pressing I Shift-Select or A Select, and you can see the anchor point position by pressing = .

When the Zap command is invoked, there is always a confirmation message presented, asking you if you really want to zap *n* characters, where *n* is some positive integer. This is to prevent accidental destruction of text, since the Z key is almost completely surrounded by other keys commonly used in the Editor.

1. Place the cursor in the blank after the word "mineralogist". Press A Select to set the anchor here. Now place the cursor in the blank after the word "mineral" in the second-to-last line of the first paragraph. Press Z. The following message should appear:

```
>WARNING: Want to Zap about 109 chars? (y/n)
```

2. Press N. The main Editor prompt comes back, and nothing has been deleted. *However, an aborted Zap does not fill the copy buffer with the text which would have been removed, as an aborted Delete does.*
3. Press Z again. Again, the prompt comes up:

```
>WARNING: Want to Zap about 109 chars? (y/n)
```

4. Press Y this time to go ahead and delete it. And, as you might expect, the deleted text goes into the copy buffer.
5. Now press M to margin your paragraph. Again, this destroys the copy buffer so now we can't get the Zapped text back.

If you answer no to the confirmation question, nothing will be deleted, and the main Editor prompt reappears. If you answer yes, the text will be deleted, but will be placed in the copy buffer, so it can be recovered. You can move the cursor to another place, and press C B to put the contents of the copy buffer into the text at the new place. Thus, large pieces of text can be moved.

When deleting enormous pieces of text, there is one other message you may see. When you do a Delete or a Zap, the deleted text is placed into the copy buffer, which is a temporary storage place in the computer's memory. If you are working on a file large enough, or deleting a large enough piece of text, there will not be a piece of memory big enough to hold a copy of all you want to delete. If this happens, you get the following message:

```
No room to copy deletion. Delete anyway? (y/n)
```

If you are actually deleting text, and have no intention of recovering out of the copy buffer, go ahead and type Y. If you are *moving* text by deletion and recovering from the copy buffer, *do not* delete the text; it will be lost. If you actually do want to move this large a chunk of text, do it in two or more pieces. In this way, the copy buffer needs less room to hold the text (it only holds one piece at a time).

Let's use the Zap command again in our example. We just destroyed a sizable portion of our file, and you may think there's not much more damage we could do. But, oh well, let's give it a shot.

Let's delete our entire file. You don't need to exit and re-enter the Editor to clear it out completely. Just jump to the beginning of the file, set the anchor there, jump to the end of the file, and Zap. In case you don't remember how to do this, it is as follows:

1. J B: Jump to the Beginning.
2. A Select: Adjust zero spaces (this sets the anchor for the Zap command).
3. J E: Jump to the End.
4. Z Y: Zap. (Are you sure?) Yes.

Voilà! The file is empty. Now let's get back a good copy of the text to continue working on.

Getting Text from Other Files

The Insert command is the most common way of creating text, but other commands are available. The Copy command allows you to copy specified text from another file.

When you Saved the file in the last section, the Editor created a file called `Mineral.TEXT`. Let's copy the entire thing into our current (empty) file.

1. Press C. The top line of your display shows:

```
>Copy: Buffer File  [<sh-sel> to leave]
```

   The Buffer option was demonstrated along with the Delete and Zap commands earlier in the chapter. Now we'll use the File option.

2. Now press F (to Copy from a File) and a new prompt appears:

```
>Copy: File[marker,marker] ?
```

   The system is requesting a file specification.

3. Type `EDIT:Mineral` and press Return. This copies the entire file called `Mineral.TEXT` from the volume `EDIT:` into the Editor.
4. The ability to copy files into the Editor is very powerful, but many times you only need part of the file. If this is the case, you can specify marker names to define exactly how much of the file you want. Again press C F; the prompt line shows:

```
>Copy: File[marker,marker] ?
```

   Type `EDIT:Mineral[mohs,]` and press Return. This copies the file called `EDIT:Mineral.TEXT`, from the marker `MOHS` to the end of the file, into the Editor.

5. Well, you probably don't want a fragmentary file there, so type C F `EDIT:Mineral[,mohs]` and press Return. This copies the same file, only this time from the beginning of the file up to the marker `MOHS`.

   If we had wanted a section out of the middle of the file, we could have specified beginning *and* ending markers: C F `Mineral[MARKER1,MARKER2]` Return.

Throwing Away All Your Work

The title of this section may be disconcerting, but there are valid reasons why you may want to exit the Editor but not store any of the work you've done. For example, say you're reading the instruction manual for a new text editor from Hewlett-Packard. During some example, you create some text you don't really want – say, you were copying files into the Editor. You don't really need the text, but you had to do the operation in order to learn how. Anyway, let's leave the Editor, throw away the damaged text, and re-enter the Editor, getting the file we stored previously.

1. Press Q E. The Editor, knowing we've made changes and have not stored them, says:

   ```
   Are you sure you want to exit without updating?
         Type Yes to Exit Without Update
         Type No  to Return to Editor
   ```

2. Press Y; we don't want our damaged text.
3. Now, you're back to the HP TechWriter Environment (or the Pascal operating system, one or the other), so reenter the HP TechWriter Editor by pressing E. You will again get a prompt like this:

```
HP TechWriter EDITOR [Rev. 1.0]   1-Jul-84

Copyright 1983 Hewlett-Packard Company.
         All rights reserved.

No workfile found.
File? (<ret> for new file, <stop> exits)
:
```

4. Type `EDIT:Mineral` Return. The Editor will find the file you stored previously, and load it in. Again, your file looks like this:

```
>TWedit:  Adjust  Copy  Delete  Find  Insert  Jump  Replace  Quit  Xchange  ?
    The German mineralogist Friedrich Mohs (1773-1839) developed a scale of
hardness in order to assign a definite figure to the values he obtained in
determining mineral hardness.  He used the scratch-hardness method,
observing the comparative ease or difficulty with which one mineral is
scratched by another.

    Another scale of hardness, called the Vickers hardness scale, was
created during World War II.  In the Vickers method of determining
hardness, a pyramidal diamond is pushed into the material being tested for
about fifteen seconds, under a specified load.  The indenter is removed,
and the surface area of the indentation is measured under a microscope.

    A graphical comparison of Mohs' and Vickers hardness numbers is shown
below.
```

In the whole example thus far, we've been dealing with paragraphs. Now, we'll see how to work with text that we don't want the Editor to format for us.

Making Tables

A commonly-used form of data is a table – a collection of words and/or numbers in which the *position* of the characters is meaningful. As you can see above in our sample paragraphs, it makes no difference to the meaning of the paragraph where on the line any particular word occurred, or how many lines there were, etc. However, in a table, the position of an item can make a big difference.

Let's make a table of data.

1. Set *program mode* in the environment. You accomplish this by pressing (S) (P). Program mode prevents the Editor from automatically margining (formatting) your text into paragraphs.
2. Press the space bar to exit the environment-setting menu.
3. Move the cursor to the beginning of the second paragraph and add this text in table form to your file. Use the (Tab) key to space over to the beginning of the second column. (We haven't told you yet what the `^^m off` does, but include it anyway.)

```
^^m off
Mohs' Hardness Scale
  1        Talc
  2        Gypsum
  3        Calcite
  4        Fluorite
  5        Apatite
  6        Orthoclase
  7        Quartz
  8        Topaz
  9        Corundum
 10        Diamond
```

4. Press (Return), (Return), and end the insertion with (Select).
5. Press (=) to move the cursor to the position of the beginning of the last insertion (the table), and press (C) (B). Another copy of the table appears, showing that inserted text, like deleted text, is placed in the copy buffer.
6. Move the cursor to somewhere in the first table, and try to margin the text by pressing (M). The machine beeps, and you get the message:

```
>ERROR: Wrong environment <space> continues.
```

Margining cannot be done while in program mode. Press the space bar to clear the error message.

While margining affects the left/right positions of words in a paragraph, there is another command which affects left/right position of *lines*. This command, called Adjust, is suited for horizontal positioning of text lines.

The next section discusses how to use the Adjust command for moving individual lines to the left, right, and center. A subsequent section will re-visit the subject of margining to show how to use additional margin-related features for detailed control over the margining process.

Formatting Text

The HP TechWriter Editor allows you to format text with the Adjust and Margin commands. We'll examine the Adjust command in this section. Text can also be formatted by inserting or deleting blanks where needed, assuming Document mode is not set. (If Document mode is set, the extra blanks will be automatically deleted.)

The Editor's Adjust command provides a means of shifting the starting, or ending, column of a line of text left or right in the file. This command helps make your tables, programs and other text more readable. To increase the aesthetics of our tables, let's center them.

1. Move the cursor to any character in the title of the first table.
2. Press A and the Adjust prompt appears:

```
>Adjust: Left Right Center <arrow keys> [<sel> to leave]
```

3. Press 2 5 ▶ (this is exactly the same as pressing ▶ twenty-five times). This will move the main title of the table twenty-five places to the right, which is about centered.
4. The Editor remembers how far and in what direction you've adjusted, so now when you press ▼ (just do it once), the next line is moved twenty-five places to the right also.
5. Here again, you can use a repeat factor. Press 2 0 ▼; this will move the following twenty lines twenty-five places to the right.
6. After using repeat factors on a vertical arrow, you can still use single-keystroke arrow keys. Press ▼ two more times to center the rest of the second table.
7. Press Select to exit the Adjust command.

The Adjust command remembers the most recent command you've given it, and every time the cursor moves to a new line, the adjustment is made, whether or not it has already been made.

1. To illustrate this, move the cursor to the first line of the first paragraph.
2. Press A to invoke the Adjust command.
3. Press 2 ▶. The first line is moved two columns to the right.
4. Press ▼ three times. The following three lines are adjusted two columns to the right also. This is nothing new.
5. Now press ▲ ▲. The two lines the cursor entered were moved two *more* columns to the right.
6. Press ▼ ▲ ▼ ▲. The two lines are again moved – every time the cursor enters the line – for a total of four more columns.
7. Press Shift-Select to try to abort the Adjust command. The Editor beeps but does not abort the command. You *cannot* terminate the Adjust command with a Shift-Select.
8. Press L to adjust the line to the left margin. Move the cursor to all lines in the first paragraph, ending up on the first line of the paragraph. Press 4 ▶. Now the paragraph has been restored.
9. Press Select to terminate the Adjust command.

As you can see, the adjust command adjusts lines according to its most recent instructions *every time it enters a new line*. The adjustments are cumulative, and not relative to the position of the lines when the Adjust command was invoked. Therefore, be sure you have exited an Adjust command before moving the cursor indiscriminately.

Not only does the HP TechWriter Editor allow you to adjust text by absolute amounts, as just shown, it also allows you to adjust the horizontal position of text in such a way that every line may be adjusted by a different amount, such as you might want in centering, or right-adjusting (producing "ragged-left") text.

1. Jump to the Beginning of the file.
2. Insert the following text (you may want to press (Caps) before and after typing the first line):

   ```
   METHODS OF DETERMINING MINERAL HARDNESS
   Freshman Geology
   Dr. Ross
   ```

   Press (Return) (Return) to make sure there is one blank line after this new text, and then press (Select).
3. Now move the cursor to somewhere in the first line and press (A). The Editor again displays this prompt:

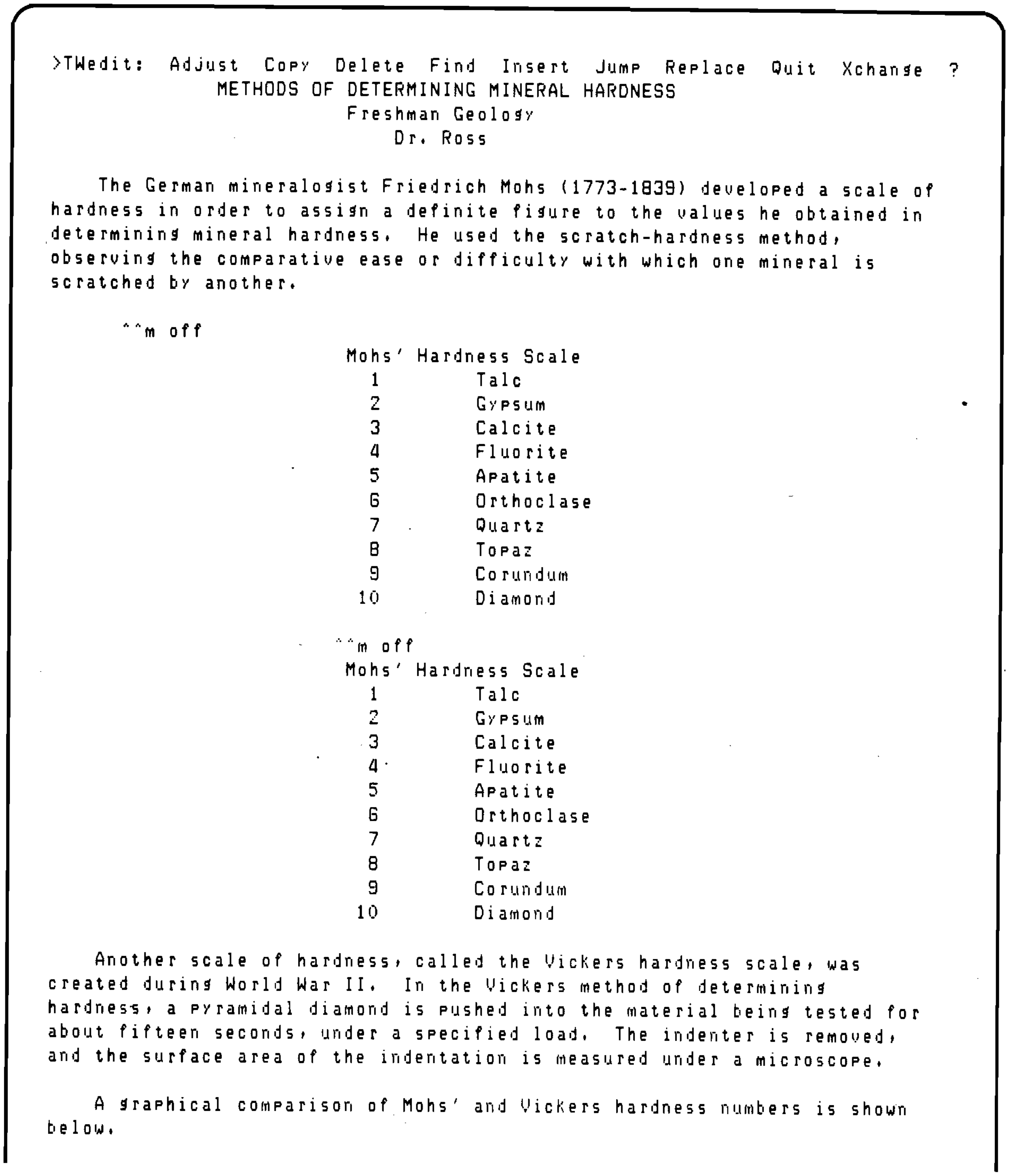
```
>Adjust: Left Right Center <arrow keys> [<sel> to leave]
```

4. Press (C). As you see, this centers the current line.
5. Press (▼). The second line is also centered. But since the second line was a different length than the first line, *it was adjusted a different distance*. Press (▼) again. The same thing occurs for the third line.

   Your "title" text now looks like this:

   ```
                  METHODS OF DETERMINING MINERAL HARDNESS
                             Freshman Geology
                                 Dr. Ross
   ```
6. Now press (R). The centering command is replaced by a right-adjust command. The professor's name is bumped up against the right margin. (This can look like it didn't work if there are trailing blanks on lines. The rightmost character is the character that is justified, regardless of what it is. If you want your text to be right justified, delete any trailing blanks.)
7. Press (▲) (▲), and the other two lines in the title are adjusted against the right margin.
8. Press (L) (▼) (▼), and the lines are left-justified.
9. Press (C) (▲) (▲) and the lines are once again centered. Accept the adjustment with (Select).

Now your file looks like this:



```
>TWedit:  Adjust  Copy  Delete  Find  Insert  Jump  Replace  Quit  Xchange  ?
                 METHODS OF DETERMINING MINERAL HARDNESS
                             Freshman Geology
                                 Dr. Ross

    The German mineralogist Friedrich Mohs (1773-1839) developed a scale of
hardness in order to assign a definite figure to the values he obtained in
determining mineral hardness.  He used the scratch-hardness method,
observing the comparative ease or difficulty with which one mineral is
scratched by another.

      ^^m off
                           Mohs' Hardness Scale
                             1        Talc
                             2        Gypsum
                             3        Calcite
                             4        Fluorite
                             5        Apatite
                             6        Orthoclase
                             7        Quartz
                             8        Topaz
                             9        Corundum
                            10        Diamond

                          ^^m off
                           Mohs' Hardness Scale
                             1        Talc
                             2        Gypsum
                             3        Calcite
                             4        Fluorite
                             5        Apatite
                             6        Orthoclase
                             7        Quartz
                             8        Topaz
                             9        Corundum
                            10        Diamond

    Another scale of hardness, called the Vickers hardness scale, was
created during World War II.  In the Vickers method of determining
hardness, a pyramidal diamond is pushed into the material being tested for
about fifteen seconds, under a specified load.  The indenter is removed,
and the surface area of the indentation is measured under a microscope.

    A graphical comparison of Mohs' and Vickers hardness numbers is shown
below.
```

Remember: when adjusting, if you make a mistake, there is no way get your text back like it was without adjusting again, and doing complementary actions. (Shift)-(Select), the old faithful "abort-this-command," merely beep. You *must* exit an Adjust command by pressing (Select).

Note

Think twice before using Adjust with a large or infinite repeat factor. This is because (Shift)-(Select), which usually aborts all changes made by a command, is not available for exiting the Adjust command. Therefore, to recover the original format of your text, you would have to Adjust it again.

If you wish to make adjustments in several non-contiguous parts of your text file, exit the Adjust command using (Select) before moving the cursor from one area to another. Otherwise you may make unwanted adjustments to your text.

After saving our most recent changes to our text, let's learn more about the global environment.

Let's Save the File One More Time...

We have just added some more text to our file, and we don't want to lose it, so press:

1. (Q): this *Q*uits the Editor,
2. (S): this *S*aves the file under the same name as before, and
2. (R): this *R*eturns to the Editor.

The Global Environment

Several times in the course of this chapter, you have been instructed, "Go into the environment" or "Set Document mode" or "Set Program mode." You may already have noticed some of the characteristics of these two "modes," but you've probably seen a lot of very mysterious stuff flash past when you did these things. Oh, what does it all mean?

The environment display that you've been using shows all global switches and special characters that the Editor uses. You saw in sections past how the settings of `Token search` and `Ignore case` affected the operation of various commands. These settings, as well as everything else mentioned in the environment display, are *global*, that is, they affect everything in the entire file. Many of the switches available in the global environment are discussed in this chapter. If you want to know more about them, or if you want to know about switches that are not discussed here, you can look them up in the *Editor Reference* chapter, which discusses all of them.

You can alter the environment at any time using the *S*et *E*nvironment command. Once you have altered the environment and saved a `.TEXT` file on a mass storage medium, that environment is stored along with the text file and is used whenever the Editor is entered with that file. When the Editor is entered with either a workfile or the name of a `.TEXT` file, the environment associated with that file is the current environment.

There is a way to alter the "default" environment parameters. That is, you can specify what the environment parameters will be when you enter the Editor without a file. Instructions for doing this are in the *Customizing the Default Environment* section in the next chapter.

Text Entry Modes

There are two text entry modes provided by the Editor. They are *Document* mode and *Program* mode.

Document Mode

In our example, when you entered the Editor with a .TEXT file that you had previously stored on a disc, the environment associated with that file came along with it. Document mode was set when you stored the file, and it was still set when you brought the file back in.

When writing paragraphs, you want the Editor to do formatting for you. Therefore, in Document mode, doing an Insert causes an automatic margining operation and the Margin command is enabled. Extra spaces will be removed and the Editor will fit as many words as possible between the margins you've selected.

1. To see how the impact of setting the environment to Document mode, press S D (for Set Document) from the main Editor prompt. The first few lines on your screen look like this:

```
>Environment: {options} <sel> or <sp> leaves

  Auto indent   False                    Command ch    ^
  Filling       True                     Underline ch  \
```

Notice that:

- The `Auto indent` switch is `False`. When you enter a new line, it will start in the left column.
- The `Filling` switch is `True`. Therefore, word-wrap takes place on insertion.

  Only when these two switches have the values shown above are you in Document mode. In Document mode, margining will occur both on insertions and when explicitly requested.

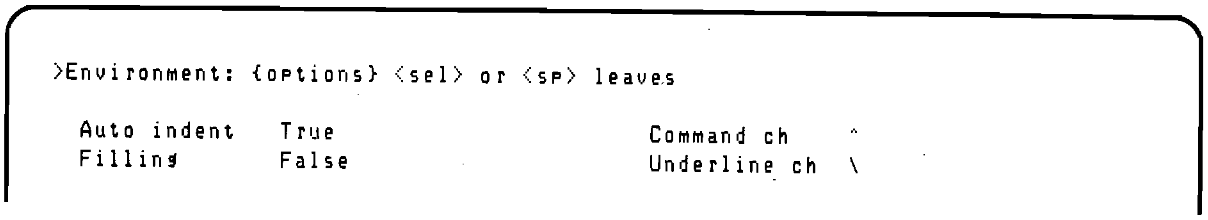
2. Press the space bar to leave.

Program Mode

Entering the Editor without a workfile or a named file automatically sets the environment to "Program" mode (unless you changed the configuration file; see *Customizing the Default Environment* in the next chapter). Program mode is optimized for writing programs, tables and other text which should not be margined, but retained as typed.

When writing programs or creating tables, you don't want the Editor to format your text for you. Program mode prohibits margining the text, so the spaces and line-endings you enter are preserved.

1. To see the impact of setting the environment to Program mode, press S P (for Set Program) from the main Editor prompt. The first few lines on your screen look like this:



```
>Environment: {options} <sel> or <sp> leaves

 Auto indent   True                  Command ch    ^
 Filling       False                 Underline ch  \
```

   Notice that:

   - The `Auto indent` switch is `True`. When you enter a new line, it will be indented the same amount as the previous line was.
   - The `Filling` switch is `False`. Therefore, word-wrap will not take place on insertion.

2. Press the space bar to leave.

Now you know about the Program and Document modes. Next, we will revisit the topic of margining with a more in-depth discussion of its features than was presented earlier in this chapter.

Margining

Margining is the process of packing words in a paragraph as tightly as possible between the margins. This causes paragraphs to look like you expect them to.

The margining operation works on a *paragraph* as its smallest unit. **A paragraph is defined by the Editor to be any text delimited by any combination of blank lines, lines having the command character as the first non-blank character in a line, or the beginning or end of a file.** If you don't keep this definition in mind, you might margin things you don't want margined.

In all the previous sections where you entered paragraphs, column 75 was used for several things (unless you are using a Model 226 computer, in which case substitute "column 45" for "column 75" in the discussion that follows):

- Column 75 was used as the right-most limit every time you margined the paragraph.
- Also, words were automatically placed on the next line if they started to extend past column 75.
- Finally, column 75 was also used as the right margin when you Adjusted the title text.

The reason column 75 was the cutoff point was that the `Right margin` parameter in the environment defaulted to 75, and you never changed it.

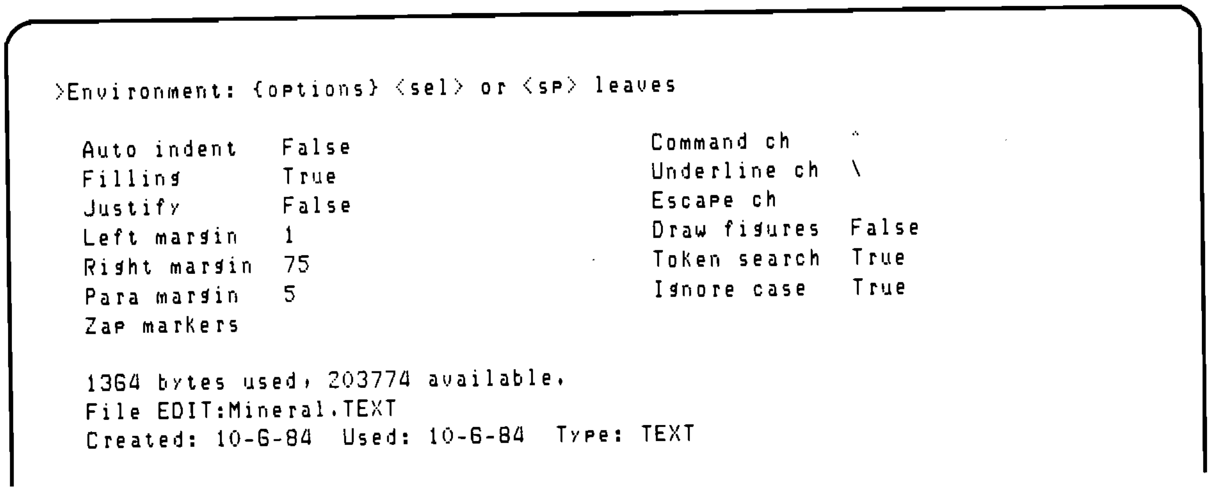
Margining takes place according to the Left, Paragraph, and Right margin settings of the environment. There are some things you need to know about these margin-related values:

- All are relative to column zero, not column one. Thus, if the left margin is 0 and the right margin is 60, your line of text is 61 columns wide, not 60.
- The paragraph margin may be equal to, greater than or smaller than the left margin. Thus, if `Para margin` is 5 and `Left margin` is 10, the first line of each paragraph will extend five spaces to the *left* of the rest of the paragraph; that is, it will be "outdented."
- The paragraph margin and the left margin must both be less than the right margin.
- The right margin can be no farther to the right than column 255.

You can modify the values in the environment menu to suit your needs.

1. Go into the environment-defining mode and set Document mode by pressing S D.

   The screen looks similar to the following:



```
>Environment: {options} <sel> or <sp> leaves

  Auto indent   False                          Command ch     ^
  Filling       True                           Underline ch   \
  Justify       False                          Escape ch
  Left margin   1                              Draw figures   False
  Right margin  75                             Token search   True
  Para margin   5                              Ignore case    True
  Zap markers

  1364 bytes used, 203774 available.
  File EDIT:Mineral.TEXT
  Created: 10-6-84  Used: 10-6-84  Type: TEXT
```

2. There are three margin parameters we want to change. First, define the `Left margin` to be zero: L 0 Return. This says that there will be no space to the left of the text.
3. Define the `Para margin` to be four: P 4 then press the space bar. This says that the indentation of the first line of a paragraph will be four spaces to the right of the rest of the lines. (For numerical values like these, the numbers may be terminated by either the space bar or Return.)
4. Define the `Right margin` to be 65: R 6 5 Return.
5. Now you're done modifying the environment, so press the space bar to exit the environment-setting menu.
6. Your text did not change when you returned from setting the environment, even though the definition of how it should look has been modified. After making sure the cursor is in the first paragraph (not in the title text), press M to execute the Margin command. The Margin command reformats the paragraph to make it fit in the global margins you just specified. Now the first paragraph looks like this:

```
    The German mineralogist Friedrich Mohs (1773-1839) developed a
scale of hardness in order to assign a definite figure to the
values he obtained in determining mineral hardness.  He used the
scratch-hardness method, observing the comparative ease or
difficulty with which one mineral is scratched by another.
```

7. Press S E. This gets you into the environment-setting menu, but *does not change any values.* That is, it does not set Document mode *or* Program mode.
8. Now set the left margin to 8, the right margin to 40, and return to your text. This would be L 8 Return R 4 0 Return, **space bar**. Margin the paragraph again (M), and it should look like this:

```
The German mineralogist Friedrich
    Mohs (1773-1839) developed a
    scale of hardness in order to
    assign a definite figure to the
    values he obtained in determining
    mineral hardness.  He used the
    scratch-hardness method,
    observing the comparative ease or
    difficulty with which one mineral
    is scratched by another.
```

9. In all of the above examples, you may have noticed that the right edge of your text is not lined up (this is called *ragged right*). If you want the right margin to be straight also, spaces must be inserted between the words on each line. This can be done easily: go into the environment-setting mode (S E) and press J T. This sets justification to true. And while you're in there, set `Left margin` to 0 and `Right margin` to 65 again. Press the space bar to leave.
10. Now margin the paragraph one more time, and it should look like this:

```
    The German mineralogist Friedrich Mohs (1773-1839) developed a
scale of hardness in order to assign  a  definite  figure  to  the
values  he obtained in determining mineral hardness.  He used the
scratch-hardness  method,  observing  the  comparative   ease   or
difficulty with which one mineral is scratched by another.
```

So, in summary, you can make paragraphs as narrow or as wide as you desire, and you can choose ragged or straight right edges for your paragraph.

11. Set the Justify parameter back to false, and we'll continue the example.

But What About Tables?

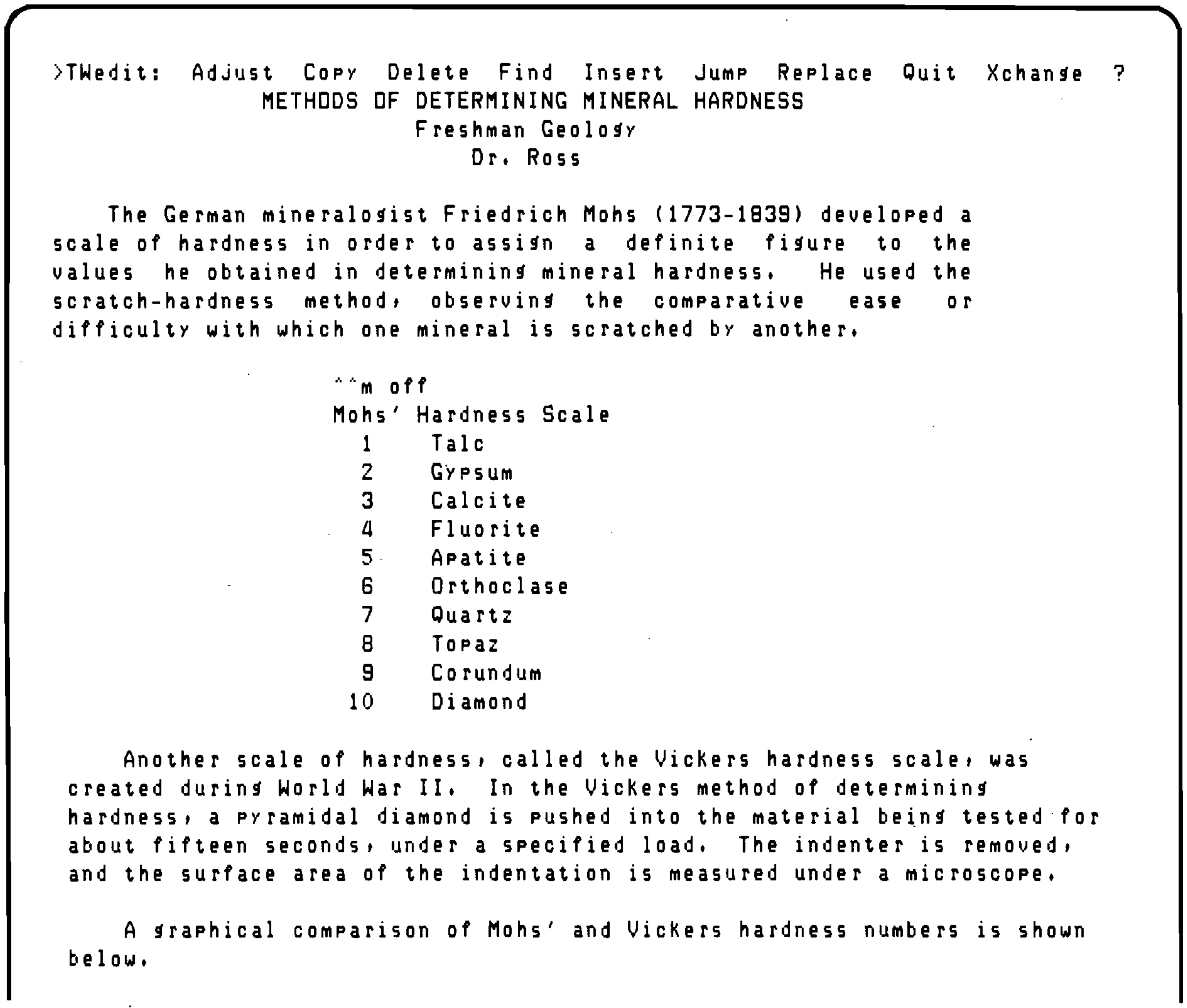
In a table, margining is definitely inadvisable, as you'll see now.

1. Remove the line from the first table which says `^^m off`.
2. Press M, and the result will be this:

```
    Mohs'  Hardness  Scale  1 Talc 2 Gypsum 3 Calcite 4 Fluorite 5
Apatite 6 Orthoclase 7 Quartz 8 Topaz 9 Corundum 10 Diamond
```

As you can see, a table in this form is more difficult to interpret. This gives you some idea of how important it is to know what your environment settings are and where the cursor is located before using the Margin or Insert command. Unless you had saved the table previously, the only way to recover your table is to use a combination of the Adjust and Insert commands to rebuild the text.

3. Using the Delete command, get rid of the disaster area you just caused, but keep the unmargined copy of the table. If you wish, you can also re-adjust the table; it is no longer centered since you changed the right margin. At the moment, then, your file should look like this:



```
>TWedit:  AdJust  Copy  Delete  Find  Insert  Jump  Replace  Quit  Xchange  ?
               METHODS OF DETERMINING MINERAL HARDNESS
                          Freshman Geology
                              Dr. Ross

   The German mineralogist Friedrich Mohs (1773-1839) developed a
scale of hardness in order to assign  a  definite  figure  to  the
values  he obtained in determining mineral hardness.  He used the
scratch-hardness  method,  observing  the  comparative   ease   or
difficulty with which one mineral is scratched by another.

                    ^^m off
                    Mohs' Hardness Scale
                      1     Talc
                      2     Gypsum
                      3     Calcite
                      4     Fluorite
                      5     Apatite
                      6     Orthoclase
                      7     Quartz
                      8     Topaz
                      9     Corundum
                     10     Diamond

    Another scale of hardness, called the Vickers hardness scale, was
 created during World War II.  In the Vickers method of determining
 hardness, a pyramidal diamond is pushed into the material being tested for
 about fifteen seconds, under a specified load.  The indenter is removed,
 and the surface area of the indentation is measured under a microscope.

    A graphical comparison of Mohs' and Vickers hardness numbers is shown
 below.
```

Now that we've mastered the Editor's ability to margin according to our specifications, let's discuss how to specify *different margining parameters for every paragraph.*

Local Margins when Formatting

In all the previous examples of the Margin command, you were using the global margins – those set in the environment. Many times, however, you will want to have certain paragraphs use different left, right, or paragraph margins. And in tables, as you saw, you don't want them to be margined at all. So, as you enter the text, you set the environment appropriately for every paragraph as you arrive there.

If life was as simple as just stated above, you would have all the capabilities you needed. You can margin with globally-defined margins, change those margin definitions for certain paragraphs, and completely avoid tables and such. But let us put forth a scenario:

You have created a large file – hundreds of paragraphs – and you have margined all the paragraphs in it, sometimes with `Justify True`, other times with `Justify False`. Some paragraphs have one set of margins, others have another set of margins. Also, there are many tables, which shouldn't be margined at all. Suddenly, you find out that a particular term you have used throughout the document is incorrect.

The fix is simple: merely replace all occurrences of the errant term with the correct term. However, since the new term is a word of a different length than the old one, every paragraph which had smooth right margins and which uses the term will have its right margins messed up after the replace operation. (The problem also occurs if right margins are ragged, but it's not always so obvious.)

Enter local margins.

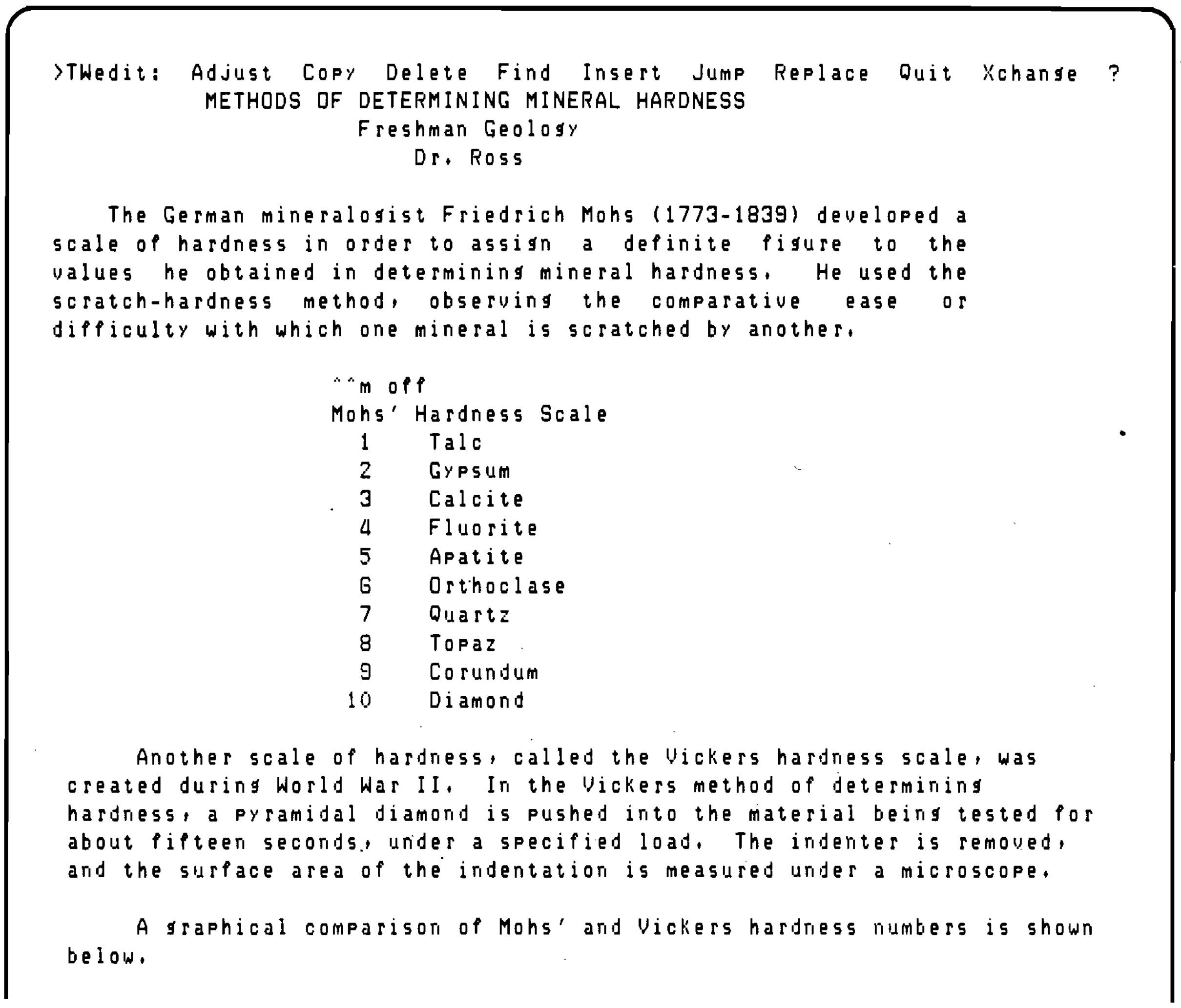
The HP TechWriter Editor has a set of commands called document commands. These are commands which are not executed from the keyboard, but are actually placed into the document as text. Document commands are identified by having two command characters as the first two non-blank characters in the line. ("Ah, ha!" you say. "That is what the `^^M off` command was!" You're right.)

The command character is defined in the environment. It is a character which is given special significance. When the first non-blank character of a line of text in the Editor is the command character, the Editor will not margin that line. Also a line which begins with the command character separates paragraphs in the same way a blank line does.

When the first *two* non-blank characters of a line of text in the Editor are both command characters, that line is considered a *document command.* A document command contains special instructions for the Editor. We'll discuss the local margin document command now and other document commands later in the manual.

The local margin document command allows you to specify local margins which apply *only to the following one paragraph*, and which may be different from the global margins.

Let's do an example. Your file should look like this currently (the whole thing won't fit on the screen at the same time):



```
>TWedit:  Adjust  Copy  Delete  Find  Insert  Jump  Replace  Quit  Xchange  ?
           METHODS OF DETERMINING MINERAL HARDNESS
                      Freshman Geology
                          Dr. Ross

       The German mineralogist Friedrich Mohs (1773-1839) developed a
scale of hardness in order to assign  a  definite  figure  to  the
values  he obtained in determining mineral hardness.  He used the
scratch-hardness  method,  observing  the  comparative   ease   or
difficulty with which one mineral is scratched by another.

                    ^^m off
                    Mohs' Hardness Scale
                      1    Talc
                      2    Gypsum
                      3    Calcite
                      4    Fluorite
                      5    Apatite
                      6    Orthoclase
                      7    Quartz
                      8    Topaz
                      9    Corundum
                     10    Diamond

      Another scale of hardness, called the Vickers hardness scale, was
 created during World War II.  In the Vickers method of determining
 hardness, a pyramidal diamond is pushed into the material being tested for
 about fifteen seconds, under a specified load.  The indenter is removed,
 and the surface area of the indentation is measured under a microscope.

      A graphical comparison of Mohs' and Vickers hardness numbers is shown
 below.
```

Assuming that the command character as defined by the environment is a caret (^), a local margin specifier might look similar to this:

```
^^M L=5,P=10,R=50,J=F
```

This command tells the Editor that the paragraph immediately following it should be margined with the left margin in column five, the paragraph margin in column ten, the right margin in column fifty, and to margin it with ragged right (justify false). The parameters L, R, P, and J may be in any order. Upper and lower case letters may be used interchangeably.

If you have paragraphs you do not wish to margin, the following document command will accomplish it:

```
^^M OFF
```

When we told you to use this command above the table, it was because the table was not supposed to be margined.

Let's use these commands in the text.

1. First, set the following values in the environment:

   ```
   Auto indent    True
   Filling        False
   Justify        True
   Left margin    0
   Right margin   65
   Para margin    0
   Command ch     ^
   ```

   Leave the environment-setting menu.
2. Go to the beginning of the file. Press (I) to invoke the Insert command, and enter the following line:

   ```
   ^^M off
   ```
   (Return) (Select)
3. Go to the beginning of the first line of the first paragraph, and insert the following line:

   ```
   ^^M L=10,P=10,R70
   ```
   (Return) (Select)

   The equals signs are optional between parameters and their values, as in the "R" parameter above.
4. Go to the beginning of the first line of the second paragraph (it starts "Another scale of hardness...), and insert:

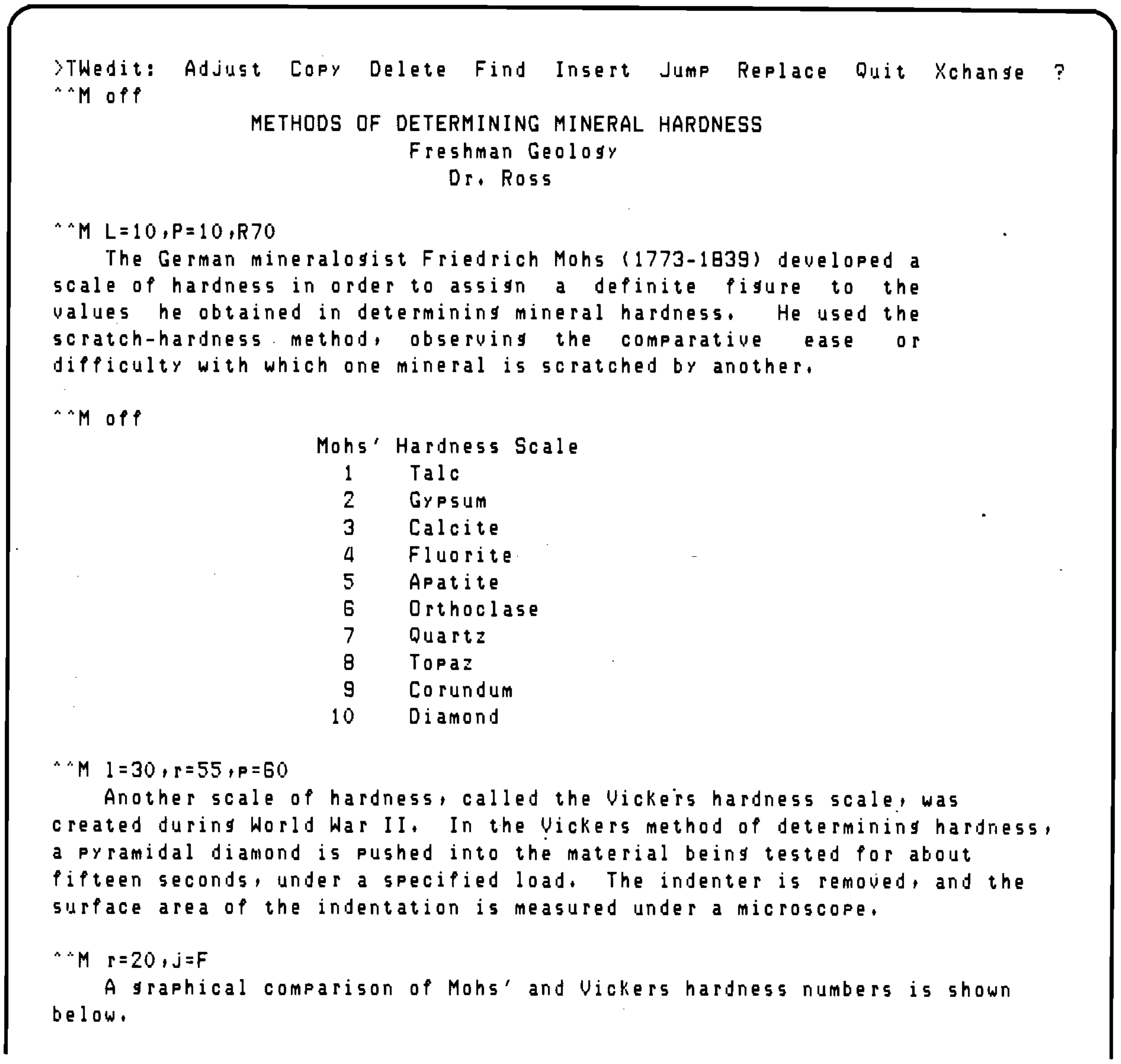
   ```
   ^^M l=30,r=55,p=60
   ```
   (Return) (Select)

   Notice that the paragraph margin has a larger value than the right margin, which is illegal (the left end of a line can't be farther right than the right end). Since one of the specifications is illegal, the Editor doesn't know what you really wanted so the global (environment) values are used for all margin values.
5. Before the final paragraph, place the following line.

   ```
   ^^M r=20,j=f
   ```
   (Return) (Select)

   Note that the global values for left and paragraph margins will be used, since they are not specified in the document command. Also, you want ragged right for this paragraph.
6. Now, although it's not necessary, we recommend that you use the Adjust command to move the document command lines to the far left. This makes them easier to see when editing a file.

Now, your text should look like this:

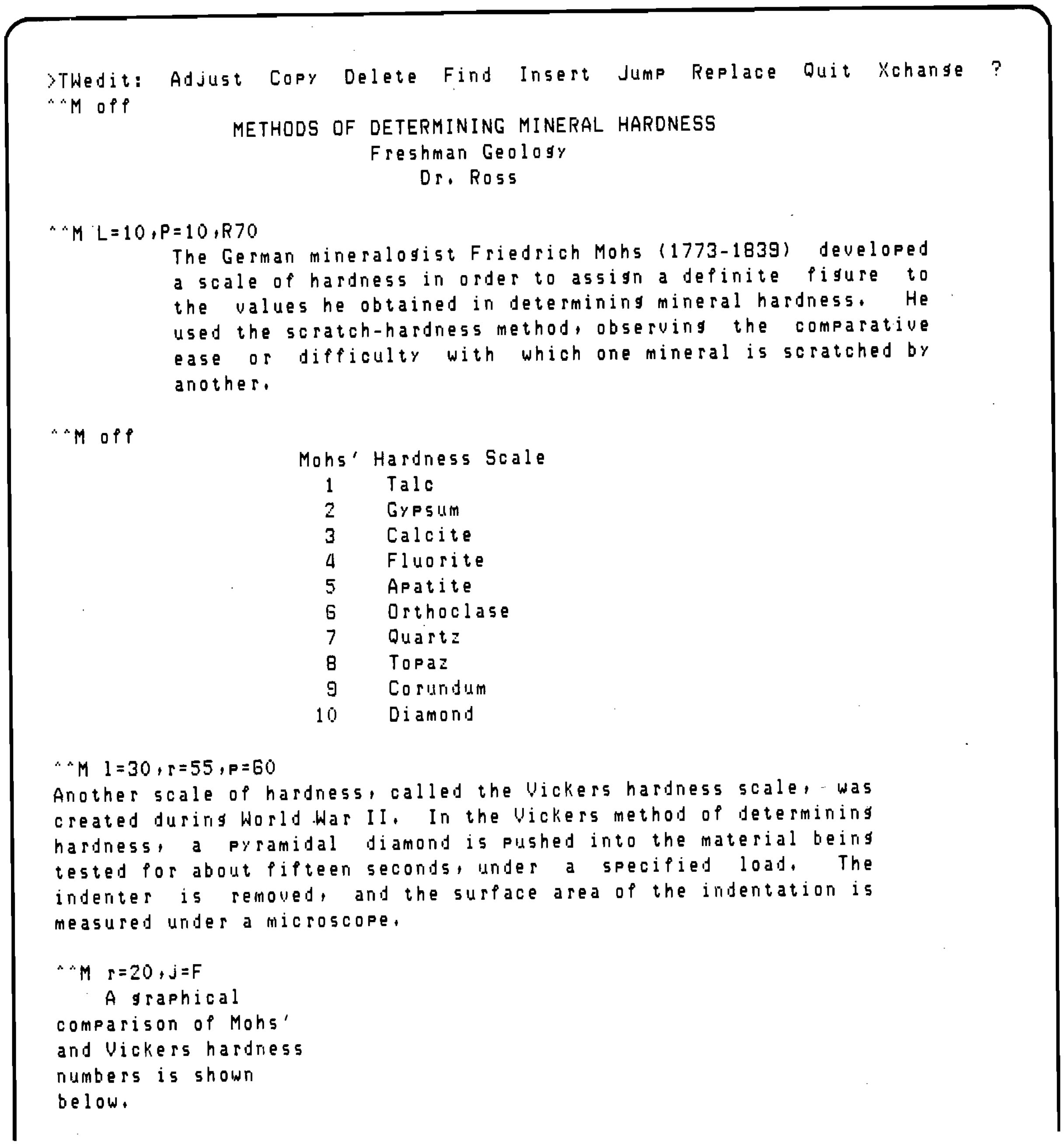


```
>TWedit:  AdJust  Copy  Delete  Find  Insert  Jump  Replace  Quit  Xchange  ?
^^M off
                METHODS OF DETERMINING MINERAL HARDNESS
                          Freshman Geology
                             Dr. Ross

^^M L=10,P=10,R70
    The German mineralogist Friedrich Mohs (1773-1839) developed a
scale of hardness in order to assign  a  definite  figure  to  the
values  he obtained in determining mineral hardness.  He used the
scratch-hardness  method,  observing  the  comparative   ease   or
difficulty with which one mineral is scratched by another.

^^M off
                    Mohs' Hardness Scale
                      1     Talc
                      2     Gypsum
                      3     Calcite
                      4     Fluorite
                      5     Apatite
                      6     Orthoclase
                      7     Quartz
                      8     Topaz
                      9     Corundum
                     10     Diamond

^^M l=30,r=55,p=60
    Another scale of hardness, called the Vickers hardness scale, was
created during World War II.  In the Vickers method of determining hardness,
a pyramidal diamond is pushed into the material being tested for about
fifteen seconds, under a specified load.  The indenter is removed, and the
surface area of the indentation is measured under a microscope.

^^M r=20,j=F
    A graphical comparison of Mohs' and Vickers hardness numbers is shown
below.
```

Now you'll see how to margin everything in the file with one command.

1. Set Document mode in the environment (S, D, space bar).
2. Press > to ensure that the direction is forward.
3. Jump to the beginning of the file.
4. Press / M. This means margin each paragraph, using local specifications (if any), in the current direction, from the cursor position to the end of the file. Of course, you can specify any repeat factor you want. If a paragraph has no local specifications, the global environment specifications will be used.

After the global margin, your file should look like this:



```
>TWedit:  Adjust  Copy  Delete  Find  Insert  Jump  Replace  Quit  Xchange  ?
^^M off
               METHODS OF DETERMINING MINERAL HARDNESS
                          Freshman Geology
                              Dr. Ross

^^M L=10,P=10,R70
          The German mineralogist Friedrich Mohs (1773-1839)  developed
          a scale of hardness in order to assign a definite  figure  to
          the  values he obtained in determining mineral hardness.   He
          used the scratch-hardness method, observing  the  comparative
          ease  or  difficulty  with  which one mineral is scratched by
          another.

^^M off
                    Mohs' Hardness Scale
                      1    Talc
                      2    Gypsum
                      3    Calcite
                      4    Fluorite
                      5    Apatite
                      6    Orthoclase
                      7    Quartz
                      8    Topaz
                      9    Corundum
                     10    Diamond

^^M l=30,r=55,p=60
Another scale of hardness, called the Vickers hardness scale, was
created during World War II.  In the Vickers method of determining
hardness,  a  pyramidal  diamond is pushed into the material being
tested for about fifteen seconds, under  a  specified  load.   The
indenter  is  removed,  and the surface area of the indentation is
measured under a microscope.

^^M r=20,j=F
    A graphical
comparison of Mohs'
and Vickers hardness
numbers is shown
below.
```

In this way, selected portions of your file, or the whole thing, can be margined with one command. At the end of a margin, no matter how many paragraphs were affected, the cursor will be left at the point it was when the margin command was initiated.

If you wish to stop a repeat-factor margin command before it finishes, you can do so by pressing Shift-Select. For example, if you want to margin the next nine paragraphs, you would press 9 M. But, little did you know, your finger wasn't precisely centered on the 9 key, and you are dismayed to see the margin telltale:

```
>Margin[90]...
```

At this point, you can press Shift-Select. The Editor checks after margining every paragraph to see if you have pressed Shift-Select. If you have, margining will stop. The cursor will be left at the beginning of the paragraph being margined at the time you pressed Shift-Select, so you can tell how far you went.

The Shift-Select must be the *first character* in the type-ahead buffer[1]; if it is not, it will not stop the margining process. If you press Shift-Select and nothing seems to happen, press CTRL-Clear line to clear out the type-ahead buffer, then press Shift-Select again. The margining will stop.

1 The type-ahead buffer is a temporary storage place for characters you type. When you press a key, that keystroke goes into the type-ahead buffer. The keystrokes remain in this temporary storage place until they're needed, at which time they're taken out and consumed by the program currently running. This means that you can answer questions before they are asked. Or, you can pile up commands in the type-ahead buffer if you type them faster than the computer executes them. This is a perfectly safe practice.

Locking Margins

So far we have seen how to set local margins which affect the next paragraph only. There is also a feature in the Editor which allows you to keep using the local margins you specified until you explicitly change them. Using the optional keyword `ON` makes the local margins you put into effect remain active until they are explicitly turned off. This is called turning on *locking margins.*

For example, if you want to turn on locking margins whose left edge is at 20 and whose right edge is at 60, you would say:

```
^^M L=20,R=60 ON
```

The keyword `ON` (again, case is ignored in keywords) causes the specified margins to be "locked on" until you explicitly change them. Any subsequent margin command will cause the locked-on margins to be changed. If you want to go back to using global margins after a section using locking margins, you would specify:

```
^^m
```

Let's try an example.

1. Delete the lines containing the local margin commands for the second-to-last and last paragraphs.
2. Above the second-to-last paragraph (the one that starts "Another scale..."), add the following line:

   ```
   ^^M l=10,p=14,r=40,j=t on
   ```

3. Jump to the beginning of the file.
4. Press [/] [M] to margin the whole file. The titles, the first paragraph, and the table will not change, but the last two paragraphs will now look like this:

   ```
   ^^M l=10,p=14,r=40,j=t on
               Another scale of  hardness,
           called   the  Vickers  hardness
           scale, was created during World
           War II.   In the Vickers method
           of   determining   hardness,  a
           pyramidal   diamond  is  pushed
           into  the material being tested
           for  about   fifteen   seconds,
           under  a  specified load.   The
           indenter is   removed,  and the
           surface area of the indentation
           is measured under a microscope.

               A  graphical  comparison of
           Mohs'  and   Vickers   hardness
           numbers is shown below.
   ```

   Observe that the last paragraph was affected by the margin command above the *previous* paragraph.

We mentioned before that the way to disable margining for a paragraph is to say `^^M off`. As you might have guessed, the way to disable margining *until further notice* is to type:

```
^^M off on
```

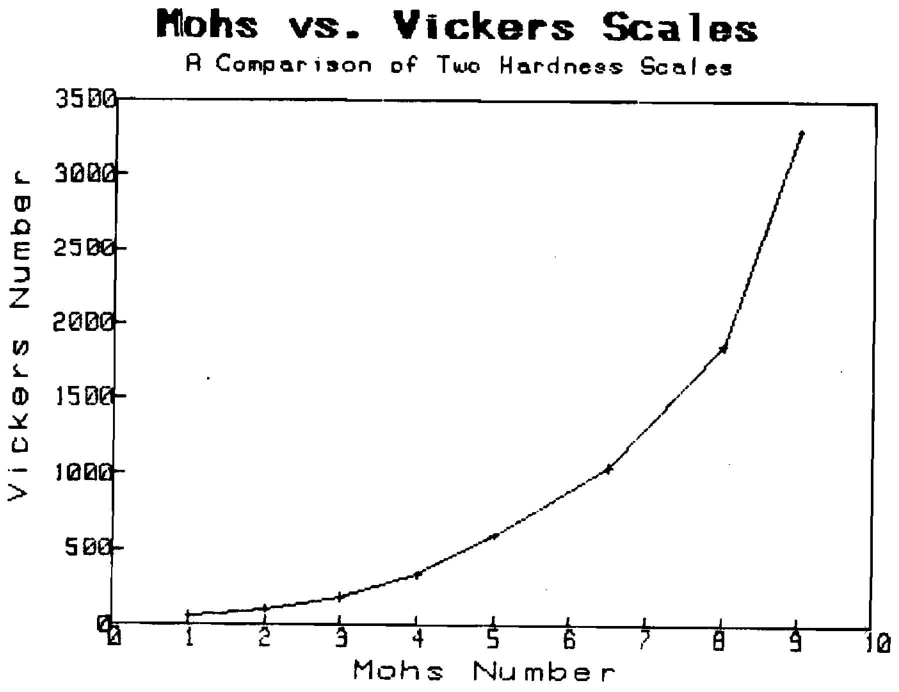
Positioning Drawings on the Page

One exciting feature of the HP TechWriter Editor is the ability to place drawings right in the text itself. We saw a little of this capability earlier in the chapter, and now we'll delve deeper.

1. Disable margining by pressing (S), (P).
2. Move the cursor to the drawing document command (^^D).
3. Insert the `LIST:` disc in the disc drive and close the door.
4. As you may remember, to get the Editor to even attempt to draw a figure on the screen, the `Draw figures` parameter in the global environment must be `True`. Go into the environment now and press (D) (T) – this will set `Draw figures` to `True`. Press the space bar to leave the global environment menu.
5. When you come back, scroll the screen until both command lines are visible, if they are not already. As soon as the entire picture area is on the screen, the file specified in the Draw command will be accessed and the picture will be drawn.

   The picture should look like this currently:

```
^^D f=LIST:mohs
```



```
^^
```

   There are sixteen blank lines between the `^^D` line and the `^^` line.

6. If no picture is drawn, and you merely get a box around the drawing limits, the Editor cannot find the file specified. Let's misspell the file name on purpose, so the Editor can't find the plot file. Change the file name "`mohs`" to "`moes`".
7. Press (V) to tell the Editor to redraw the screen. Notice how, after looking around a bit, the Editor places a box around the drawing limits.
8. Now change the file name back to "`mohs`" so we can continue the example.
9. Press (V) again and the Editor will redraw the picture.

The source of the drawings can be any graphics program which can send its plotting commands to a file; contact your local Hewlett-Packard Sales Representative to decide which one(s) are most suited to your needs. As a matter of fact, your own programs (either written in HP Pascal or in BASIC 3.0) which draw special-purpose pictures can also route their plotting commands to files that HP TechWriter Editor can assimilate.

All plot files must be processed by the HP TechWriter Picture Processor before they can be accessed by the Editor. This was discussed in the *Picture Processor Tutorial* section of the last chapter.

Drawing Limits

A figure being drawn by the HP TechWriter Editor is drawn within the confines of "drawing limits," a left edge, a right edge, a top edge, and a bottom edge.

- The drawing's left-right limits are specified by left and right margins. As in margining, the left and right margins may be either global or local. Local margins for a drawing command are specified by switches in the `^^D` command.
- The bottom and top drawing limits are determined by the number of blank lines between the `^^D` document command and the next document command, i.e., the next double-command-character line. Typically, but not necessarily, an empty command (e.g., `^^`) is used.

1. Go the end of your file, and insert two blank lines (press (Return) three times to do this) and the following text. Be sure to leave one blank line between the document command and its terminator (the `^^`).

   ```
   ^^D l=10,r=10

   ^^(Select)
   ```

   A small box should appear in column 10 of your screen.

   Left/right margins are specified in character cell units. The left edge of the picture is the left edge of the specified left margin position and the right edge of the picture is the right edge of the specified right margin position. This is why you get a box instead of a line when you specify both the left and the right margins to be 10.
2. Insert three more blank lines between the drawing command and its terminator. A tall, skinny rectangle should be drawn on your screen. The height of a drawing is solely dependent upon the number of lines between the Draw figure command and its terminator.
3. Change the ^^D command you just inserted so that it reads:

   ```
   ^^D l=10,r=9
   ```

   Press (V) again so the picture is redrawn. A wide rectangle should be on your screen. If the local left margin is to the right of the local right margin, both are ignored, and the global margins are used. If only one local margin is specified, and it is on the wrong side of the complementary global margin, it also is ignored, and the global margins are used.

4. Change the ^^D command you just inserted so that it reads:

   ```
   ^^D l=10,r=90
   ```

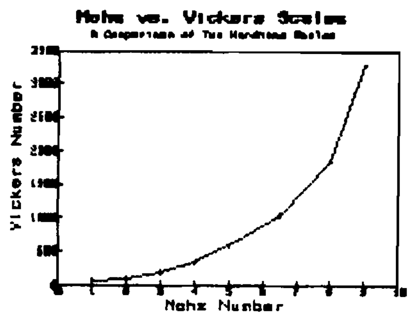
   Press (V) again so the picture is redrawn. The rectangle should extend to the right edge of your screen. If you specify a right margin that is farther to the right than is possible on your display, the maximum value for the display is used: 49 for the 50-column Model 226, and 79 for the 80-column Models 216, 217, 220, and 236 (column numbering starts from zero, not one).
5. Delete the eight lines we have just inserted in this example.

The above examples drew boxes around the edges of the specified drawing area. Now we'll go back and work with the picture we used earlier.

1. Modify your Draw command so that it looks like this:

   ```
   ^^D f=LIST:mohs,l=30,r=50
   ```

   After modifying a line which defines the limits of a drawing, the drawing may or may not be re-drawn, depending on the keystrokes you hit during the modification. However, you can always get a picture re-drawn by either scrolling the drawing area off the screen and then back on, or just by pressing (V) (Verify). After being re-drawn, the resulting picture would be:



2. You can move the picture horizontally by adjusting the left and right margin specifiers. Change the left and right margin specifiers to 10 and 30, respectively, and re-display the picture. It looks the same as before, but it has moved 20 columns to the left.

Adjusting Pictures in the Drawing Area

Pictures are normally centered in the drawing area. If a picture is drawn which does not fill the entire drawing area, there is some room left over either on the right and the left, or the top and the bottom. You can specify the justification of the picture within the drawing limits. The options to the Draw command which do this are:

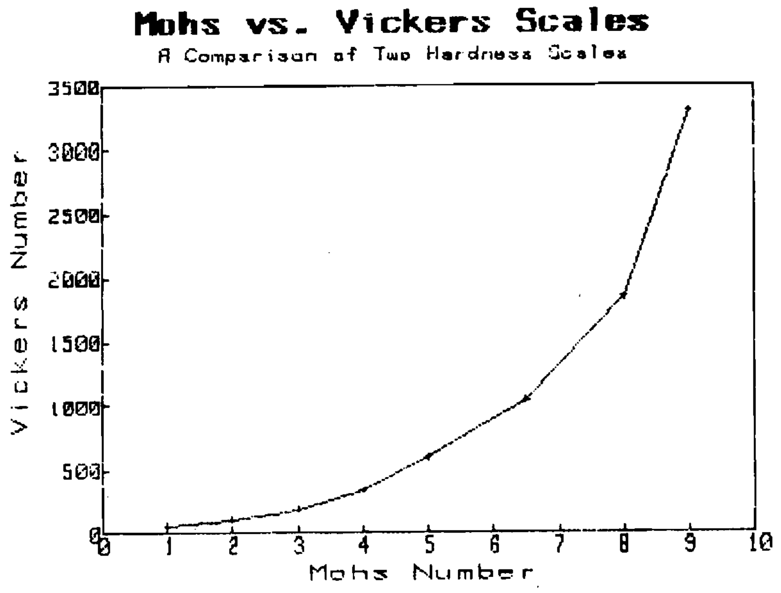
- To horizontally justify the picture, specify H=L, H=C, or H=R to left-, center-, or right-justify the picture within the drawing limits, respectively.
- To vertically justify the picture, specify V=B, V=C, or V=T to bottom-, center-, or top-justify the picture within the drawing limits, respectively.

1. Modify your drawing command so it looks like this:

```
^^D f=LIST:mohs,h=l
```

This tells the Editor to push the picture to the left edge of the drawing limits. The picture should look like this when it is redrawn.

```
^^D f=LIST:mohs,h=l
```



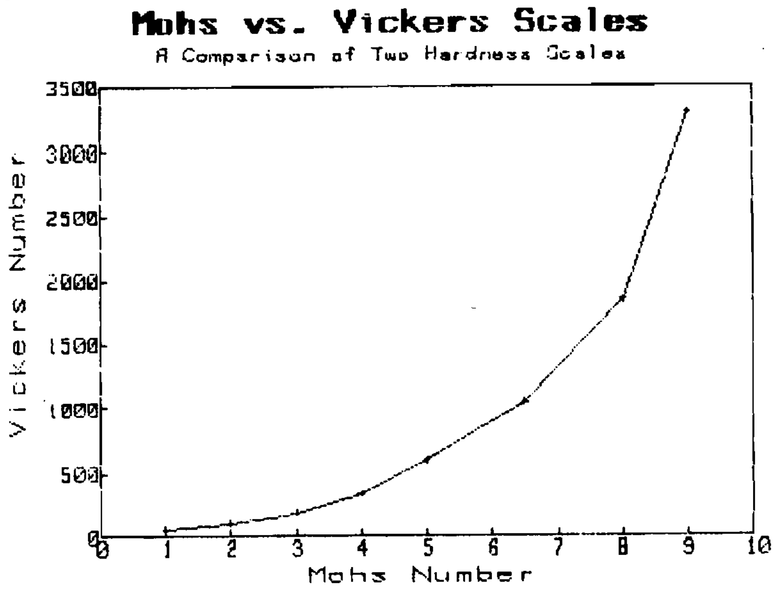
```
^^
```

2. Modify your drawing command so it looks like this:

```
^^D f=LIST:mohs,h=r
```

This tells the Editor to push the picture to the right edge of the drawing limits. The picture should look like this when it is redrawn.

```
^^D f=LIST:mohs,h=r
```



```
^^
```

3. Remove the `h=r` from the drawing command. The next time the picture is redrawn, it will once again be centered.

The picture-justification capability is especially useful when sending drawings to printers whose dot size, resolution, and width:height ratio are different from those of the screen. This way, the Editor knows what to do with any left-over space which was not present in the CRT representation of the figure.

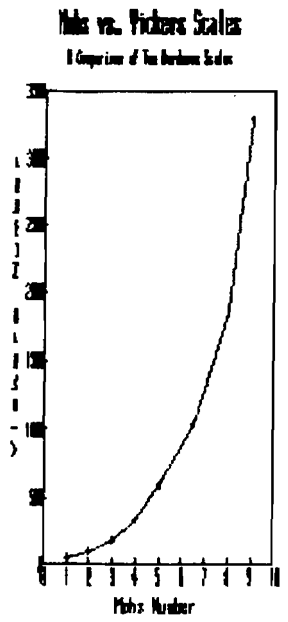
Picture Options

In the above example, you altered the width of the drawing limits without altering the height, but the rectangle in the picture remained the same shape. It may seem like it should have gotten tall and skinny, since the width changed but the height did not. The reason the rectangle did not change shape is that the HP TechWriter Editor normally draws figures *isotropically;* that is, one unit in the horizontal (X) direction is the same as one unit in the vertical (Y) direction.

1. To get the drawing to be drawn *anisotropically,* that is, to fill up the entire area, you specify the `s` parameter in the Draw figures command. Modify the drawing command so that it looks like this:

   ```
   ^^D f=LIST:mohs;l=30,r=50,S
   ```

   This results in the picture looking like the following:

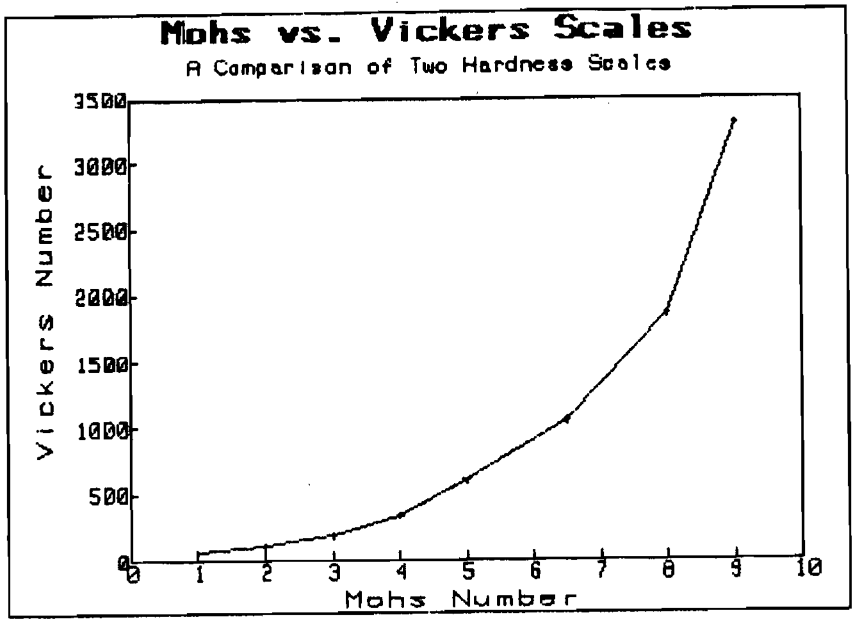


The "s" specifier causes the picture to be "stretched" to fill all the available space.

2. Let's draw a frame around the picture. The "B" specifier causes a box to be drawn around the current drawing limits, whether global or local. Modify the drawing command so that it looks like this:

```
^^D f=LIST:mohs,b
```

The modified command results in the picture looking like this:



When a box is drawn around a picture, the picture is reduced in size very slightly, so the box doesn't cover up any of the picture.

3. When dealing with a very complex drawing, the drawing time may be more than you want to wait. So, you can abort the drawing process. Press (V), followed immediately by (Shift)-(Select). The drawing process will be stopped at the point where you press (Shift)-(Select).

   As in aborting the multi-paragraph margin command, the (Shift)-(Select) must be the first thing in the type-ahead buffer. If it is not, it will have no effect. If the (Shift)-(Select) does nothing, press (CTRL)-(Clear line) to empty the type-ahead buffer, then press (Shift)-(Select) again. The drawing will be immediately aborted.

During the time when the figure is being drawn, the cursor will be placed at the end of the Draw figure command currently being processed. After the image is completed, the cursor will be placed in the position it was before the picture started being drawn.

Archiving Your File

Here is the last place you'll store a copy of this example text file. This time, however, we'll use the `Archive` option again, to learn more about how it works.

1. Make sure the `EDIT:` disc is on-line.
2. Press Q.

```
>Quit:
     Update the workfile and leave
     Exit without updating
     Return to the editor without updating
     Write to a file name and return
     Archive file, write new file EDIT:Mineral.TEXT
     Save as file new file EDIT:Mineral.TEXT
     Overwrite as file EDIT:Mineral.TEXT
```

3. Press A. This archives the old file, and creates a new file. When you select the `archive` option, the file that currently has the name shown in `new file` has an `@` appended to its name. The current Editor contents are then written to the file name shown. In this example, `EDIT:Mineral.TEXT` gets renamed to `EDIT:Mineral@.TEXT`, and the current Editor contents are stored in `EDIT:Mineral.TEXT`. Any existing file named `EDIT:Mineral@.TEXT` is deleted *without warning or comment.*

   As you may remember, a file named `EDIT:Mineral@.TEXT` was created when you used the `A` option at the end of the first Editor tutorial. That file was deleted when you pressed `A`. An archive file is *only* for temporary storage of the last copy of your file. If you wish to save the contents of the archive file, you should use the `Change` command from the `Filer` program right away to give the archive file a "safer" name.
4. Now press E (for the Exit option) and you will leave the Editor.

This brings us to the end of the Editor tutorials. The next chapter, *Fully Utilizing HP TechWriter*, will deal with the many output options available through HP TechWriter. The List command from the Editor and the Lister will be covered, as will the document commands that they use. Some of the techniques you can use to achieve "fancy" printing are explained. The *Customizing the Default Environment* section is also included in the next chapter.

Fully Utilizing HP TechWriter

Chapter 5

Introduction

This chapter presents information for both the novice and the experienced text editor·user – it is neither tutorial in nature as the preceding chapters were, nor is it purely reference oriented as the last chapters in the manual are. The first three sections of this chapter describe ways to use the HP TechWriter features that relate to controlling the appearance of your printed output. The final section of the chapter describes the means with which to personalize your Editor environment defaults.

Document Commands for Printing Effects

This section presents the document commands that affect the appearance of your printed output. To use any of these commands, simply insert the command at the appropriate place in your text.

Paging

When printing a large document, there are three capabilities dealing with paging which are very important. It is oftentimes necessary to be able to:

- Start at the top of a new page at any point in your document;
- Specify the page number at any point;
- Start at the top of a new page if you don't have enough space left on the current page.

These three capabilities are provided with the document command "P". To set a new page unconditionally, just use the simplest form of the command:

```
^^P
```

To request a new page conditionally depending on how much space is left on the current page, use the "C" option. For example, when doing the table in the *More About Editing* chapter, the Editor needed eleven lines in which to print the table. If the current page had only had five lines left on it, the table would have been printed across the perforation, resulting in part of the table on each of two consecutive pages. You could certainly avoid this by preceding the table with an unconditional new page request (`^^P`), but then you could run into the opposite problem: if only three lines had been printed on the page, you wouldn't want a new page to be generated; there would be *plenty* of room for the table.

The conditional option on the page command covers such cases. If the conditional option is used, the Editor will start a new page only if there is not enough space left on the current page. How much space is needed is specified by using the "C" option on the page command. The paging command for the table in our earlier example looks like this:

```
^^P C=11
```

This says: "If there are less than eleven lines left on the current page, go to the top of a new page." Or, more succinctly put, "I need eleven lines. If I have them, fine. If I don't, start a new page."

The other capability, that of setting the page number, is accomplished by using the N option on the page command:

```
^^P n=46
```

This says, "Start a new page immediately and make it be page 46."

If you do not want a new page started, but want to change the page number for *this* page, specify a conditional pagefeed if there are less than zero lines left on the page:

```
^^P N=46,C=0
```

Since there are never less than zero lines left, you will change the page number, but never get a pagefeed with this kind of command.

Underlining

It is often useful when writing documents to be able to underline parts of your text. The `Underline ch` parameter in the global environment lets you specify a character which will start or stop underlining text when your document is being printed out. This parameter may be any character as long as it is *not* one of the following:

- A control character (ASCII code 0 through 31),
- ASCII character 127 (the "del" or "smudge"),
- The same character as the `Control` or `Escape` character in the environment.

However, it is usually not desirable to use a letter of the alphabet or a commonly-used symbol of punctuation for your underline character.

If you do not want to use underlining, you may define the underline character to be blank. A blank tells the system *there is no underline character.*

In our example text, let's underline the title of the table, using the default underline character, the backslash (\). Text which looks like this on your screen:

```
\Mohs' Hardness Scale\
   1    Talc
   2    Gypsum
   3    Calcite
   4    Fluorite
   5    Apatite
   6    Orthoclase
   7    Quartz
   8    Topaz
   9    Corundum
  10    Diamond
```

results in this when printed:

```
Mohs' Hardness Scale
  1    Talc
  2    Gypsum
  3    Calcite
  4    Fluorite
  5    Apatite
  6    Orthoclase
  7    Quartz
  8    Topaz
  9    Corundum
 10    Diamond
```

When the Editor encounters the first underline character in the text, it tells the printer to start underlining. Text will continue to be underlined until the Editor encounters the next underline character at which time it will tell the printer to stop underlining. The next underline character causes the printer to start underlining again, and so on. The underline character itself is just a switch and is never printed out.

Because underline characters don't take any space in your printed document, they are not counted during margining operations. Therefore, a line containing underline characters may appear to extend further past the right margin than it should when the text is margined. This is especially noticeable if you are using a justified right margin.

Underlining will not be done when the `Printer type` in the list environment is `UNKNOWN` (see the *The List Command* later in this chapter).

Skipping Lines in the Document

If you want to skip several lines in a document, you can just enter that many blank lines in the document. However, this is not always desirable. For example, when you need a large number of lines, it can become tedious to count them.

There is a document command which allows you to skip as many lines as desired, up to a maximum of the number of lines remaining on the page. The command is called "Jump." If you want a twelve-line blank space in your output, specify the jump command in this way:

```
^^J N=12
```

The equals sign specifies that the space is to be in one contiguous block on one page. Thus, this command (`^^J N=12`) could skip up to 23 lines: if it were encountered when there were only eleven lines left on the page, it would go to the top of a new page – *then skip the twelve* – in order to get all twelve lines in one block.

If you want twelve blank lines in your output and you don't care whether or not they are broken across a page boundary, use the following command in your file:

```
^^J N+12
```

This command leaves *exactly* twelve blank lines in your output. You could wind up with five blank lines at the bottom of one page, and seven blank lines at the top of the next, but the Editor will skip exactly twelve lines.

Another way to use the Jump document command is to jump to the bottom of the page *minus* N lines. For example, if you want the next five lines in your text file to be placed on the last five lines of the document's page, you could say:

```
^^J N-5
```

where the minus sign (the "-") indicates that the text goes at the bottom of the page. One application of this is to place text at the bottom of a user-defined form. For example, many forms have a small area at the bottom which says something like "Office use only." Suppose you want two lines of text at the bottom of the form and suppose you want a horizontal line separating the main body of the form from the bottom section (underlining a series of blanks creates a horizontal line). You could say:

```
^^J n-3
\                                                       \
\OFFICE USE ONLY\           Expiration date: \             \
Comments:
```

When the above commands are encountered, lines are skipped until there are three left on the page. Next, the three lines following the `^^J` command are printed. Then, of course, the page is full, so a new page is begun.

If the number of lines you request at the bottom of the page is greater than the number of lines left on that page, the Editor will do the operation on the following page. This could conceivably result in the "bottom text" being the only thing on a page.

There is another optional parameter on the `^^J` command. You can specify a condition which, if true, causes the requested number of blank lines to be output.

The `C` option causes a condition to be tested before any blank lines are output. If the condition is true, then the number of lines is skipped, *up to a maximum of starting a new page.*

Consider the following command:

```
^^J C=15,N=12
```

When this command is encountered while the document is being listed, these tests are used, and the following results are output:

1. Is the number of lines left on the page *less than* fifteen?
   a. If so, then go to the top of a new page.
   b. If not, skip twelve lines in one contiguous chunk.

The command

```
^^J C=7
```

tells the Editor: If there are seven lines left on the page, skip seven lines. In other words, if `N` is missing, skip `C` lines.

Specifying a jump command with N equal to zero does nothing.

One restriction applies to the Jump document command. You *cannot* use it to skip space which will be filled with a drawing. The reason for this is that the drawing command is terminated by *any* double-command-character line, including a `^^J` command.

Double-Spacing, Etc.

Many times, you may want to have a document printed with double-spacing or triple-spacing. In the HP TechWriter Editor the `^^S` command allows you to specify *n*-spacing, where *n* may be any integer between zero and the number of lines on the page. For single-spacing, set *n* equal to 1, for double-spacing, set *n* equal to 2, etc. Overprinting, that is, printing something and then printing something else on the same line, maybe in the same columns, is accomplished by setting *n* equal to zero.

Let's look at some examples:

`^^S`	Set single-spacing.
`^^S N=1`	Set single-spacing.
`^^s n3`	Set triple-spacing.
`^^s n=12`	Set duo-decuple spacing.
`^^s n=0`	Set zero-spacing (overprinting).

While the uses for single-, double-, and triple-spacing are obvious, it might not be quite so obvious what you would use zero-spacing for. After all, who could read it if you just wrote something, and then wrote over the top of it? There are two reasons why you might want to do overwriting:

- You can create pseudo-bold printing by overwriting the same text several times; and
- You can create your own characters by combining two or more characters.

Both of these features are described in later sections of this chapter.

Specifying Page Footers

You can specify a footer, or *tag*, to be placed at the bottom of each page with the `^^T` command.

The tag command is a two-line command. The first line contains the command itself, and the second line specifies the contents of the tag. For example, if you want every page to have the text, "`An Introduction to Geometry`" placed at the bottom, you could specify the following tag command:

```
^^T
An Introduction to Geometry
```

There are three values that are commonly used in footer tags which you typically do not know at the time the document is prepared: the current date, the current time, and the current page number. You can specify these in the tag by using `<date>`, `<time>`, and `<page>`, respectively. These tokens must have all lower case characters or all upper case characters, and may not include blanks.

The order in which the components of the date are printed can be specified in the configuration file. You may specify any order for month, day, and year, and you can specify the delimiter which is to be printed between them. See the section *Customizing the Default Environment* later in this chapter for more information.

In this section, we will assume the default order `MDY/`. The date would then be printed in the form *mM/dD/YY* (e.g., 4/20/83), where the lower case *m* and the lower case *d* are only used if necessary. That is, leading zeroes are not included, except on the year number. The time is printed in the form *HH:MM* (e.g., 12:15), and it is printed in 24-hour format. All the numbers in the time have leading zeros, if appropriate. For example 2:01 p.m. is represented as *14:01*. The page number is just the bare number itself.

For example, if you are printing a document at 9:05 a.m. on October 6, 1984, the tag command:

```
^^T
My Summer Vacation (First draft:  <date> - <time>) Page <page>
```

would produce the following tag on page six:

```
My Summer Vacation (First draft: 10/6/84 - 09:05) Page 6
```

The date and the time are the system values at the point when the first `L` is pressed from the Editor or Lister. Thus, although it takes a certain amount of time for your printer to print each page, the time value in the tag will *not* change from page to page.

If you do not want tags at the bottom of the document's pages, you can either:

- Set the Listing parameter `Include Tag` to `False` (we'll cover this shortly), or
- Define a blank tag – merely make a blank line follow the the `^^T` command. For example:

  `^^T`
  <blank line>

The default tag is the following:

```
^^T
                                        Page <page>
```

The default tag is centered as shown. If you specify a tag value with a `^^T` command, it will be printed exactly as you typed it. If you want it to be centered, use the Adjust Center command on the text line. You can adjust it on the line so it looks any way you want it to.

If you specify a page tag more than once, only the most recently specified one is in effect each time a page tag is printed; the previous definitions are replaced.

It is important to remember that the `^^T` command requires two lines to completely define. Consider the following portion of a file:

```
^^T
^^P
```

The line immediately following a `^^T` command is the tag definition, so the `^^P` will be printed as the tag at the bottom of every page. Since the "^^P" is used as a tag and not a command, the Editor will not start a new page here.

Noting Table-of-Contents Entries

The `^^C` document command allows you to specify an entry to be included in the table of contents. Like the tag command, it requires two lines. Again, like the tag command, the first line is the command itself, and the second line is the text to be used.

A table of contents is generated only by the Lister program, not by the List command in the Editor.

To define the title for the table of contents itself, include the `T` option in the Contents command. This title will be included in the actual listing of the document, unless you specify otherwise with the `OFF` keyword. For example:

```
^^C T off
                  A Treatise on World Energy Usage
```

will define "A Treatise on World Energy Usage" to be the title of the table of contents, but it will *not* appear in the actual listing, while

```
^^C T
                  A Treatise on World Energy Usage
```

will define "A Treatise on World Energy Usage" to be the title of the table of contents, and the text line *will* also appear in the actual listing.

Notice that the title for a table of contents was centered in the definition line. It will be printed exactly as it appears in the line, so you can position it any way you desire.

If you specify more than one table of contents title, only the last one will be used.

For a "regular" table of contents entry, use the form of the Contents command with no options. An example of this simplest form of the Contents command follows. If there are no parameters included in the command, the entry (defined on the subsequent line) will appear in the actual listing, in addition to being included in the table of contents. This would be used for things like section headings.

```
^^C
Section I:  Bituminous Coal
```

If you want the entry to appear in the table of contents, but *not* appear in the listing, here again, you can type `OFF`:

```
^^C off
Section II:  Anthracite Coal
```

The previous commands result in the following table of contents (the page numbers may vary):

```
               A Treatise on World Energy Usage


                      Table of Contents


  Topic                                              Page
Section I:  Bituminous Coal.......................... 1
Section II:  Anthracite Coal.........................15
```

You can specify the two column labels, "`Topic`" and "`Page`", as well as the header, "`Table of Contents`", by altering the contents of the configuration file (see the section *Customizing the Default Environment* later in this chapter for further information). These column labels are printed at the right and left margins of the first file in the Lister program file pool. If this file is not a `.TEXT` file, the margins set by the configuration file are used.

The Lister uses the first tag it encounters while processing the files in the file pool for the table of contents tag. If it does not encounter a tag command, the default tag from the configuration file is used. If you want special margins or a special tag for your table of contents, you may want to make a file with those margins containing nothing but the desired tag to use as the first file in your file pool.

`<page>` tokens in tables of contents tags are printed as lower case Roman numerals.

If you want a blank line left in your table of contents, use a `^^C` command followed by a blank line.

Disabling Sections of a Listing

On occasion, you may want sections of text to remain in your file, even though you do not want them to be included in a listing of the file this particular time. Another document command, the `^^L` command, allows you to do this.

By default, all document commands are interpreted and all text is printed. To turn off a particular section of text, so as to not print it during the listing, type:

```
^^L off
```

All text following this command will be excluded from the listing, and all document commands will be **ignored** until listing is re-enabled by the following command:

```
^^L
```

The List Command

The double-command-character document commands just discussed wouldn't be as useful were it not for the List command, which is one of the options from the main Editor and Lister prompt. Before we discuss this command, let's find out what actually goes on when you list a document.

What Happens When I List My Document?

There are many operations which take place when you list a document. The most obvious operation is that the text file is read one line (of up to 255 characters) at a time, and it is sent to the printer. But before it goes, there can be a lot of processing taking place first.

- Underline characters are noted. If underlining is currently off, and an underline character is encountered, the command to turn underlining on is sent to the printer. Similarly, if underlining is currently on, and an underline character is encountered, the command to turn underlining off is sent to the printer.
- Escape sequences are noted. An escape sequence is an escape character followed by three digits which comprise a number less than 256. If a valid escape sequence occurs, the character value of the number specified is sent to the printer at that point in the line of text. Thus, all the special abilities of your printer are at your disposal. See the section *Printer Escape Sequences*, later in this chapter, for more information.
- If the first non-blank character of a line is a command character, and the following character is *not*, the command character is replaced with a blank before the line is output.
- If the first two non-blank characters on a line are adjacent command characters, this line is a document command, and it is processed accordingly.
- If the listing is taking place from the Editor with `Include Commands` (one of the List parameters) `True`, and an error occurs in a document command, an error message is printed at that time. *This is the only time when an error message is given for document commands.*

The Editor and Lister have almost identical List command options. When in an Editor session, pressing `L` gives you the following menu:

```
>List setup: {options}; <L lists>, <sel> or <sp> leaves

Number of lines per page:  60              Include Tag:          True
First page number:         1                  "    Commands:     False
Top lines skipped:         0
                                           Draw figures:         True
                                           Continuous form:      True
Printer type:              2934A
Output spool file - PRINTER:
```

The Lister has all of these parameters and two others. The additional parameters are discussed later in this section.

You may change the values of the various parameters the same way you set parameters in the Set Environment command: press the capitalized letter(s) of the option name, then type the parameter value. For example, to define `Top lines skipped` to 3, you would press T 3 Return. To define `Include Tag` to be `False`, press I T F.

On those parameters which only need a single character to complete the definition (such as `Draw figures`), the parameter entry is finished as soon as you press the appropriate character. On `Printer type`, `Output spool file`, and those parameters which require numbers, press Return to finish the entry.

Following is a discussion of the parameters available from the list setup menu.

Positioning the Page of Text on the Paper

These options dictate how the text is positioned on the page of paper. You can control where the text falls for the whole document.

How Many Lines Are There on a Page?

`Number of lines per page` defines the number of lines there will be on a page before the Editor or Lister tell the printer to go to the top of a new page. Some lines may be skipped at the beginning of a page, and some more at the end. The number specified includes all skipped and printed lines. The value cannot be less than the `Top line on page` parameter, as this would result in nothing being printed on every page. If you attempt to specify an illegal value, you will be asked to re-enter it.

Positioning the Top of the Page

The `Top lines skipped` parameter defines the number of lines skipped at the top of each page. On some printers, if text is printed on the first line of a page, it looks too close to the top. If `Top lines skipped` is 3, there will be three blank lines output at the top of each page, and the fourth line will be the first one with text on it.

The value of `Top lines skipped` cannot be greater than the `Number of lines per page` value. If you attempt to specify an illegal value, you will be asked to redefine it.

Page Numbering

The `First page number` parameter specifies the starting page number for the first page to be printed. This page number will be overridden by document commands in the file which change the page number, such as `^^P n=34`.

Optional Output

You may choose whether or not you want the page tag, document commands, and pictures in the file to be printed with the file.

There is a parameter for each of these which simply dictates whether or not to print that entity when the document is listed. These three parameters have true/false values, and may be set in any combination.

Printing the Page Tag

The `Include Tag` parameter dictates whether or not the page tag is included as the last line of each page. If it is `True`, the page tag defined in your file or the configuration file is used. If there is none, the value "`Page <page>`" is printed at the bottom of each page. This may be overridden at any time by using an embedded tag command (`^^T`).

Printing the Document Commands Themselves

If `Include Commands` is set to `True`, the document commands will be listed, then interpreted as usual. In addition, in the Editor, if any errors are encountered in a document command an error message will be printed before the listing continues. The error message indicates both the type of error and where on the offending line it was caused. If `Include Commands` is set to `False`, the document commands will not be listed, and document command errors will not be reported.

Printing Drawings

If `Draw figures` is `True`, figures referenced in your file will be printed; if it is `False`, they will not. Even when making a hard-copy listing of a figure, the figure must fit entirely on the screen, because it is drawn there before it is printed.

If you wish to have color plots, or a higher-quality plot than your printer's raster image affords, you can set `Draw figures` to `False`, print the text of your file, and then plot (with your plotter) high-quality and/or colored drawings in the spaces left for the drawings. You plot the picture with the program which originally created the drawing.

Other List Parameters

Where Should I Send the Document?

The `Output spool file` defines where the file will go when it is being listed. If you have a local printer connected to your computer and you specify `PRINTER:` (the default), the listing will go to the device specified by address 701, which means: select code 7, device address 1. This cannot be modified unless you modify the `CTABLE` program, which you can do only if you have the Pascal operating system. (See the section *Special Configurations* in the *Pascal User's Manual* for information on this.)

The value in the `Output spool file` parameter may be a file name, in which case a file will be created, and the material to be printed will be "listed" to the new file. This is useful if you want to see what a listing would look like when actually printed; for example, to see how the page breaks split paragraphs, figures, or tables. It is important to set `Draw figures` to `False` when doing this, however, because the escape sequences and bit patterns of a graphics image would wreak havoc with the Editor as you are browsing through the "printed" file.

How Should I Send the Document?

The `Printer type` parameter defines the type of printer which will be used. This is necessary because many of the capabilities used by the different printers are invoked differently. Also, picture positioning is based on printer type.

When you press `P`, the following prompt appears below the other parameters:

```
**Legal printers:
    UNKNOWN, 2631G, 2671G, 2673A, 9876A,
    ThinkJet, 2932A, 2933A, 2934A, 82906A
```

Type the name of the printer you have. If you have a printer which is not in the list, select the name of the printer which most resembles your printer, or select UNKNOWN. If you have an alpha (text-only) printer, be sure to set the Draw figures parameter to False. For unsupported printers underlining may not work and may result in extra characters appearing in your document. In this case select UNKNOWN.

The figure drawing capability may not work on printers not supported by HP TechWriter. If your pictures aren't drawn (or are drawn incorrectly), set the Draw figures parameter to False.

Single-Page Printing
`Continuous form` is almost always `True`. "Continuous form" means that successive pages are connected together and feed automatically. Most printers normally print this way. An exception is the *ThinkJet* printer when it is in single-sheet operation. In this case, you must take the printed sheet out, insert a blank sheet, and then tell the computer to continue printing. In this case, `Continuous form` would be `False`.

If `Continuous form` is `False` – that is, you are printing one page at a time – this message appears after every page is printed:

```
>Enter space when ready to continue.
```

After inserting a fresh piece of paper, press the space bar and the listing continues.

The Lister List Command

The Lister program is almost identical to the Editor List command. The prompt is slightly different. When in a Lister session, pressing (L) gives you the following menu:

```
>List environment:  {options};  <L lists>;  <sel> or <sp> leaves

Number of lines per page:   60              Include Tag:             True
First page number:          1                  "    Commands:        False
Top lines skipped:          0               tAble of contents:       False
Begin new page/reset page                   Draw figures             True
 number for each file:      False           Continuous form:         True
Printer type:               2934A
Output spool file - PRINTER:
```

The Lister has two more parameters than the Editor's List command: `tAble of contents`, and `Begin new page/reset page number for each file`. Both parameters have true/false values.

Table of Contents

If `tAble of contents` is `True`, the Lister will create a table of contents after listing all files, using all the `^^C` commands it finds in the files. The title for the table of contents comes from a `^^C T` command.

The "Topic", "Page" and "Table of Contents" headings for the table of contents come from the configuration file. See *Customizing the Default Environment*, later in this chapter for information on how to change these.

Multiple Small Documents

Sometimes, you may want a "document" printed that contains files with little logical relationship to one another. A listing for such a document might look as if you had listed each file, one at a time, from the Editor's List command. But the Lister allows you to make these small independent listings easily.

The `Begin new page/reset page number for each file` was used in the example earlier. If it is true, the Lister will start every file at the top of a page, and page numbering will start over for each file. Also, if listing was left deactivated by an `^^L off` document command in one file, listing will again be turned on at the start of the next file. If `Begin new page/reset page number for each file` is false, an `^^L off` command will remain in effect until an `^^L` command is encountered to reactivate the listing.

As its name implies, the `Begin new page/reset page number for each file` parameter causes two actions to occur if it is true, and no actions to occur if it is false. These operations are separable, however; you can begin a new page *without* resetting the page number, and vice versa.

If you want a new page at the beginning of a file, but you do not want the page numbering to start over, define `Begin new page/reset page number for each file` to `False`, and put a `^^P` (unconditional new page) command as the first line in the file.

If you want page numbering to start over, but you do not want a page eject, set `Begin new page/reset page number for each file` to `False`, and have a `^^P N=1,C=0` command as the first line of your file.

Starting the Listing

When you press L from within the List menu, the listing starts. A message is displayed indicating the listing is being made:

```
>Listing ...
```

The actual process of listing on an SRM system does not commence until you exit the List command by pressing the space bar, or until you send a listing to a new `Output spool file`.

The parameters defined in the List setup menu are part of the file's environment, and are stored with the file in files of type `TEXT` and in Lister program list files. This means that when you load the file again, or use that list file again, these parameters will be remembered and you won't have to redefine them.

Delayed Printing

You can list to a printer by pressing L L from within the Editor or the Lister. A printer expects to see ASCII data, and both programs use a special internal format which is not ASCII. Thus, the List command does a translation before sending data to the printer.

But since the List command puts out ASCII and the printer expects ASCII data, why can't you detour the data into a holding place comprised of an ASCII file? You can. This is a very useful feature if you have only occasional access to a printer.

When you enter the List menu in the Editor or the Lister, the `Output spool file` parameter is originally defined to be `PRINTER:`. If you define this to be some file name, e.g., `VOLUME:File.ASC`, you can send the data supposedly bound for the printer to the file instead. Later, you can enter the Filer and copy the file using the `Translate` command to the printer. All special capabilities will still work: overprinting, underlining, drawing figures, etc.

If you have many listing files that you wish to send to the printer, you can use wildcard characters to specify a set of files from the Filer:

```
Translate what file?
```

You would specify `MyVol:=.ASC,PRINTER:`. This would copy all the `.ASC` files in the volume `MyVol:` to the printer. Of course, if you are connected to an SRM system, you could send your listing files to the SRM printer with a command something like this: T `MyVol:=.ASC,#5:/LP/$`.

Fancy Output

This section discusses several of the ways you can use HP TechWriter features to create special effects in your printed document. It includes examples of forms, side-by-side text and graphics output, and some of the special effects available using overprinting. It also shows how the Editor escape character enables you to fully utilize all of your printer's capabilities.

Designing Forms

Quite often, it is desirable to make a form of some type, where people fill in the blanks. The underlining capability of the HP TechWriter Editor does an excellent job of this. Underlining a series of blanks results in a line which looks like you are supposed to fill it in. When you are designing a form like this, *make sure margining is disabled.* Do this either with a local margin command (as below) or by setting program mode in the environment. If you margin the text, all the series of blanks will be collapsed.

Let's design a Reader Reply Card, like many magazines have. You will need a place for the reader to enter name, address, zip code, and phone number. Assuming the underline character is the backslash (\), examine the following form:

```
^^m off on
Name: \                                                 \

Address: \                                              \

City: \                  \  State: \          \  Zip: \     \

Phone: (\    \)\   \-\      \
^^m
```

The above form definition results in the following when listed:

```
Name: ___________________________________________________

Address: ________________________________________________

City: ___________________  State: __________  Zip: ______

Phone: (____)____-______
```

Boxes Around Text

One use of the fact that a box is drawn around the drawing limits when no file is specified (or found) is to surround some text with a box. Consider the following text:

```
^^M l=14 p=14 r=57 j=t on
^^D l=0 r=71
                                WARNING

           Do not,  under any circumstances, attempt to
           force the bullpin to jump the hevershaft, as
           this will  cause  innumerable  repercussions
           along  the  entire  expanse  of multiordinal
           reality, concentrated most  heavily  in  the
           loci of any receiving nexus located near the
           hub of the local temporal continuum.

^^M
```

This file would create the following output on a 2934A printer:

```
                      WARNING

Do not,  under any circumstances, attempt to
force the bullpin to jump the hevershaft, as
this will  cause  innumerable  repercussions
along  the  entire  expanse  of multiordinal
reality, concentrated most  heavily  in  the
loci of any receiving nexus located near the
hub of the local temporal continuum.
```

Due to the different ratios of alpha to graphics resolutions on various CRT and printer combinations, you may find it necessary to experiment slightly with the boundaries of your drawing in order to achieve the desired results.

Overprinting

Using the ^^S embedded command, you can cause multiple line spacing between printed lines of text. But you can also specify line spacing to be zero. Skipping zero lines between two printed lines of text results in what is called *overprinting*. One of the most obvious uses of overprinting is to make part or all of a line bold by printing sections of the same text over and over. You are not limited to one layer of overprint; you may have as many as you desire.

Consider the following section of a file:

```
^^s n=0
This should get bolder and bolder and bolder and bolder and bolder.
                           bolder and bolder and bolder and bolder.
                                      bolder and bolder and bolder.
                                                 bolder and bolder.
                                                            bolder.
^^s
```

The above file prints out like this on an HP 2934A printer. Other printers' output would look similar, though not identical.

```
This should get bolder and bolder and bolder and bolder and bolder.
```

For two consecutive overprinted lines, you would type:

```
^^s n=0
This line has an overprinted word.
                 overprinted
^^s n=0
This does, too.
           too
^^s
```

Note that the second spacing command specifies the same spacing as is already in effect. The command is not superfluous, however. If the second command had not been there, the result would have been one quadruply-printed line, rather than two doubly-printed lines.

User-Defined Characters

Another use for the overprinting capabilities is to create your own characters by overprinting various combinations of characters. Consider the following text:

```
^^s n=0
if X<Y and o>0, then is Y=+Z?
    _      |_-           /_
^^s
```

The above text results in this:

```
if X≤Y and ɸ≥Θ, then is Y≠±Z?
```

Notice the character combinations. For instance,

- The less-than sign (<) and the underscore (_) combine to create a "less-than-or-equal to" sign,
- The lowercase "o" and the vertical bar combine to create the Greek letter "phi",
- The uppercase "O" and the hyphen combine to create the Greek letter "theta",

and so on.

The table below suggests several character combinations you might find useful. Note that some combinations will look good on some printers and not as good on other printers, since printers' character sets and resolutions are different. You may want to experiment to find what works best for your own printer.

Suggested Character Combinations

Characters to Combine		Resultant Character
less than (<)	underscore (_)	Less than or equal to
greater than (>)	underscore (_)	Greater than or equal to
equals (=)	slash (/)	Not equal to
equals (=)	tilde (~)	Approximately equals
plus (+)	underscore (_)	Plus-or-minus
lowercase A (a)	apostrophe (')	Acute-accented "a". This works with other letters, too.
lowercase A (a)	grave (`)	Grave-accented "a". This works with other letters, too.
lowercase O (o)	lowercase X (x)	Bullet. To make it darker, you can overstrike more characters.
uppercase O (O)	minus sign (−)	Theta. Depending on your printer, zero may be better than "O".
lowercase Y (y)	caret (^)	Y hat (a statistical nomenclature). Other letters may be used as well.
uppercase O (O)	slash (/)	Null set

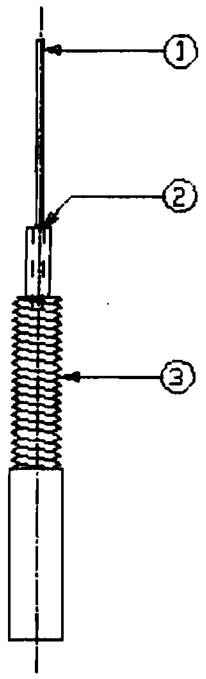
Another way to create user-defined characters is to send the first character, a back space, then the second character. For example, if the Editor escape character is the grave accent, and you want to make a ≠, you could place `=`008/` in your file. The `` `008 `` is a back space because the ASCII code for back space is 8 (see the section *Printer Escape Sequences*, later in this chapter, and the appendix on ASCII codes).

Side-by-Side Text and Graphics

This section talks about the effects that can be achieved by using the side-by-side text and graphics feature of HP TechWriter.

The following shows a typical example of side-by-side text and graphics, as the data was entered in the file, followed by its output on a 2934A printer:

```
^^M l=30 p=30 r=72 ON
^^D f=mech l=10 r=28

                                    At left is shown the first version of the
                                    75-1276B electrode.  Item 1 indicates the
                                    costly platinum (10% rhodium) wire
                                    (diameter = 0.020 inch).  This wire was
                                    selected because of its inert properties.
                                    Item 2 shows the location of the critical
                                    joint with the main body of the electrode.
                                    It is here where indications of the problem
                                    were found.  After careful study of the
                                    corrosive debris, it was determined that
                                    the plating on the exterior of the
                                    electrode was abrading during installation
                                    of the electrode into the main assembly.
                                    By changing the thread profile (item 3) on
                                    the main body, manufacturing was able to
                                    completely eliminate the abrasion.
                                    Subsequent testing has shown that with this
                                    change the life of the electrode has been
                                    increased by 500%.

^^M
```



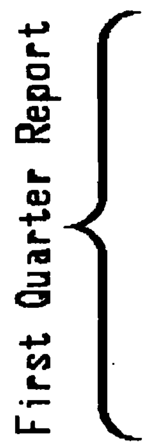
```
At left is shown the first version of the
75-1276B electrode.  Item 1 indicates the
costly platinum (10% rhodium)  wire
(diameter = 0.020 inch).  This wire was
selected because of its inert properties.
Item 2 shows the location of the critical
joint with the main body of the electrode.
It is here where indications of the problem
were found.  After careful study of the
corrosive debris, it was determined that
the plating on the exterior of the
electrode was abrading during installation
of the electrode into the main assembly.
By changing the thread profile (item 3) on
the main body, manufacturing was able to
completely eliminate the abrasion.
Subsequent testing has shown that with this
change the life of the electrode has been
increased by 500%.
```

Some rather unconventional effects achieved by this same feature are shown in the next two examples.

If you have a specific portion of text that doesn't change and you want to write it in a way that isn't available with your printer, you can *draw* the text and include it as a picture in your document. Suppose you are making a report for your company's Accounting department, discussing the fiscal year's first quarter. You could use a special graphics font from a graphics editor program to produce a picture and then include it in your report as shown below:

```
^^M l=10,p=15,r=65,j=t ON
^^D f=Figs:FirstQtr,l=0,r=10, S

          The  stock  market seemed to react favorably to our
      sale of 68,000 shares of stock.  Our  fears  that  this
      action  would cause an undesirable response in the other
      electronics companies' behaviour appear to be unfounded.
      Our shares have gained appreciably in  the  marketplace,
      and  interest  by  the press has been more than we dared
      hope for.

          Stock prices have gone up  an  unprecedented  $6.35
      the  first quarter of this fiscal year, for a percentage
      gain  of  5.62%.   Stockholders  have  expressed   much
      pleasure at our newly-instated policy which led to these
      values.

^^M
```

The result, as printed on a 2934A appears like:



```
     The  stock  market seemed to react favorably to our
sale of 68,000 shares of stock.  Our  fears  that  this
action  would cause an undesirable response in the other
electronics companies' behaviour appear to be unfounded.
Our shares have gained appreciably in  the  marketplace,
and  interest  by  the press has been more than we dared
hope for.

     Stock prices have gone up  an  unprecedented  $6.35
the  first quarter of this fiscal year, for a percentage
gain  of  5.62%.   Stockholders  have  expressed   much
pleasure at our newly-instated policy which led to these
values.
```

In this example, the *Stretch* option on the drawing command is used to make the figure extend to the limits of its borders. The only restriction on picture size is that it can never be larger than the screen. Since the picture size is specified by text lines and columns in the document, you don't have to alter the picture (or the program that created it) when you change the amount of text beside it.

Here is another example of what can be done with side-by-side text and graphics. The following text was used to produce the 2934A printer output below:

```
^^M l=6,p=6,r=62, ON
^^D f=Gothic0,l=0,r=7,h=1, S
     nce upon a time, long ago, there lived an old woman in
     her cottage at the edge of the forest.  No one quite knew
     just how long she had lived there, but everyone knew she
^^M l=0,p=0,r=62
was very old.  Even the oldest people of the town said that she
had been there as long as they could remember. Then one day,
```

```
     nce upon a time, long ago, there lived an old woman in
     her cottage at the edge of the forest.  No one quite knew
     just how long she had lived there, but everyone knew she
was very old.  Even the oldest people of the town said that she
had been there as long as they could remember.  Then one day,
```

Note that in each of the examples, a locking margin command was used so it could affect the "paragraphs" after the drawing command.

Printer Escape Sequences

Many printers provide special extended features such as math character sets and bold printing, that are not directly supported by HP TechWriter. However, HP TechWriter does permit you to access these special features by allowing you to embed printer escape sequences in your text. The escape sequences are in coded form, as required by your particular printer.

Because every printer has many different features, you will need to look in your printer manual to find out what features your printer has available. We will provide one example of what you can do on one of the HP TechWriter supported printers, below.

In the following discussion, there are three terms that need to be defined to avoid confusion:

- **Escape character** refers to the ASCII character number 27. This is shown as ␛ in the example below.
- **Editor escape character** refers to the user-definable character which is set from the global environment menu.
- **Escape sequence** refers to an Editor escape character followed by three digits. The digits must define a number less than 256. This is used to send special characters to the printer.

If you want to send an escape sequence to your printer, you need to define an Editor escape character in the global environment. The Editor escape character must be different from the control character and the underline character, but, it can be blank if you don't need one.

An occurrence of the Editor escape character followed by three digits specifies the ASCII code of the desired character. For example, assuming the Editor escape character is the grave accent (`), you would send a carriage return as `013, since the carriage return is character number thirteen in the ASCII table.

As an example, suppose you want to print the formula for the area of a circle: $A = \pi r^2$. Neither the Greek letter π nor the superscript 2 characters are directly accessible from the Editor. However, by sending control characters to the printer, *all* the capabilities of the printer are accessible.

Here is the text, followed by the result it produced on an HP 2934A printer:

```
A=`027)0M`014P`015r`0142`015
```

$A=\pi r^2$

Note the order of the characters sent:

1. Send, in the default character set, "`A=`".
2. Send the command to define the Math character set as the alternate character set: "`Ec)0M`". This makes the Math character set accessible by a mere "shift-out."
3. Send the shift-out (ASCII code 14).
4. Send the upper case "P". This is interpreted, since we're in the Math set, to be a π.
5. Send a shift-in to return the printer to the default character set. We need to do this so we can print the "r".
6. Send the "r". This will be printed as an "r" because we are currently in the default character set.
7. Send the shift-out because we need to print something else in the Math character set. The Math set is already the alternate set, so we do not need to send the command to make it so again.
8. Send the digit "2". This is interpreted in the Math set to be a superscript 2.
9. Send the shift-in to return the printer to the default character set for subsequent printing.

Whenever you use a escape-character-three-digit sequence to define a character, it is conceptually "collapsed" during margining. That is, a line containing one Escape escape character will extend four spaces farther to the right than it would had the Editor escape character not been there. This is because, much like the underline character, the escape-character-plus-three-digit sequences are not considered to be characters during margining.

There are 256 characters which may be formed by using the Editor escape character, and they range from 000 to 255. The three digits are in base ten, and any errors in the number cause the escape sequence not to be recognized. Examples of errant strings follow (the grave accent is assumed to be the Editor escape character):

`` `268 ``	The number is larger than 255.
`` `377 ``	Number is decimal, not octal; thus $n>255$.
`` `27A ``	"A" is not a digit.

There are three things to be aware of when dealing with escape sequences:

- Margins will stick out four columns per Editor escape character farther than they "should," whether or not the sequence is legal, and
- An errant string will be printed exactly as it appears on the screen and *no* ASCII character will be generated.
- Lines to be printed cannot exceed 255 characters in length including all escape sequence and underline characters.

Illustrating Program Listings

The HP TechWriter Editor is a document editor, but is also useful for writing programs (see the *Pascal 3.0 User's Guide* for more information on programming features). With little extra work, you can also use HP TechWriter to illustrate your program listings.

If you have the Pascal operating system, this section can be helpful in creating well documented program listings. If you do not have the Pascal operating system for your computer, this section will be purely academic.

The first thing you should do is set the program mode in the environment, so margining does not take place. Then type your program as desired.

Consider the following source code segment:

```
PROCEDURE PrintValues(CRT:boolean);

{  This takes the current values of the average age of the
specimens in array "Age[1..MaxSpecimen]" and prints them on
either the CRT or the printer.

The diagram below shows the data structure used to describe
the specimens:                                               }
{
^^D f=DataStruc,L=0,R=79,B




^^}
      .
      .
      .
```

When listing this portion of the source file, the following things happen:

1. The procedure statement and introductory remarks are printed.
2. The opening brace ("{") above the drawing document command (^^D) is printed. This starts a comment (according to the Pascal compiler), so when compiling this source file, the drawing command is ignored.
3. A drawing is brought into the listing and drawn in the blank lines. In this case, the drawing is a diagram of one of the data structures used in the program. It could also have contained:
   a. The procedure name in large letters, so when flipping through a listing of a very large program, the procedure names "jump out" at you.
   b. A flow chart, showing the logic flow of the routine.
   c. A block diagram, showing relationships between this and other parts of the program. This might include a "You Are Here" message so the reader could see where in the logic flow the following routine resides.

   The double-command-character line terminates the drawing, and the closing brace terminates the Pascal comment.

If you list the source code file with the HP TechWriter Editor or Lister, you will get a printout of the source file with all included pictures. If you want a compiler listing of the source file with the pictures, compile to a listing file, bring that file into the Editor, remove the line numbers from the document commands to make the command characters (^^ in this case) the first characters on the line, and list it with the List command.

Customizing the Default Environment

It was mentioned previously that when you enter the Editor without having specified a file name and without having a workfile on the system volume, there is a set of default environment settings. If these default settings are not what you would like, you can specify your own default values. Every time you enter the Editor without a file thereafter, your settings will be used. These settings will also used by the Lister program as the initial values in the list environment.

This customization is possible through the use of the *configuration file.* The file must be named `EDITORC.ASC`, and, as its name implies, it is an ASCII file. It must reside in the volume where the Editor itself resides, as specified by the *What* command from the main command prompt of the HP TechWriter Environment or the Pascal operating system.

Delivered Configuration File

You must have a configuration file because it contains the codeword and the Editor will not store files without the codeword. The values shown below are the default values that the respective parameters will have if you do not specify them in your configuration file.

To create a configuration file, use the format shown below. This file already exists on your `EDIT:` disc.

```
CODEWORD           xxxxxxxxxxxxxxxx

SEauto             True
SEfilling          False
SEjustify          False
SEleft             1
SEparagraph        5
SEcommand          ^
SEunderline        \
SEescape
SEdraw             False
SEtoken            False
SEignore           False

LPnumber           60
LPfirst            1
LPtop              0
LPprinter          2934A
LPoutput           PRINTER:
LPincludetag       True
LPincludecommands  False
LPdraw             True
LPcont             True
LPtag                                Page <page>
LPdate             MDY/
LPbegin            False
LPtable            False

LRheader                            Table of Contents
LRtopicTopic
LRpagePage
```

Configuration File Parameters

The entries of the configuration file are broken up into four categories:

CODEWORD	CODEWORD is in a class by itself. Its value is obtained by following the instructions on the security certificate delivered with your HP TechWriter system. If you operate the Editor without a correct codeword, saving your work on file will not be possible. See the section *Installing Your Codeword* in the *Protecting Yourself* chapter, for more information about codeword installation. It should be noted that, although it may be inconvenient, the Editor will work normally without a configuration file (or if the configuration file does not contain a codeword), **if** you enter the correct codeword when the missing codeword display is shown on the CRT when the Editor begins executing.
SE<*parameter name*>:	These are parameters which are defined in the global environment, which is entered by pressing S E. See the *Set* command in the *Editor Reference* chapter for more details.
LP<*parameter name*>:	These are parameters which are defined in the list environment, which is entered by pressing L. See the *List* command in the *Editor* or *Lister Reference* chapters for more details.
LR<*parameter name*>:	These are parameters which are *only* utilized by the Lister program. LRheader is the header, or title, of the table of contents; LRtopic is the title of the left column in the table of contents (where the text is); and LRpage is the title of the right column in the table of contents (where the page numbers are).

Configuration File Parameter Rules

All of the entries in a configuration file must have the parameter name *left-justified* in the file; there can be no leading blanks. Also, all entries must be spelled exactly as spelled above. The parameter names may be upper case or lower case. There may be one or more spaces between the parameter name and its value. The values have several sets of requirements, depending on their types.

Only a limited number of characters are processed for the true/false, numeric, and single-character parameters. You may put comments out to the right of these type of values. You *may not* put comments out to the right of text values (like LPtag) because *all* the text on the line (including your "comment") is used as the parameter value. Similarly, if you specify a blank value ("there is none") for the underline character or the Editor escape character, you cannot have a comment to the right, because processing continues until it reaches a character. Thus, the first character of your comment would become the underline or Editor escape character.

If a parameter name is misspelled, the specified value will be ignored. The Editor also ignores values if they are in error. Legal values for SE parameters are defined in the *Set* section of the *Editor Reference* chapter in the paragraphs relating to the particular parameter. Legal values for LP parameters are defined in the *List* section of the *Editor Reference* and *Lister Reference* chapters in the paragraphs relating to the particular parameter.

Before reading the configuration file, the Editor sets all of these parameters to the values shown in the file above. Therefore, all parameters have a "current value" when the configuration file is read (this term is used in the explanations below).

Only parameters whose default values you wish to change need to be included in the configuration file. Except for CODEWORD, none of the entries is required, and their order is generally unimportant (some exceptions are noted under *Restrictions*, below).

Configuration File Parameter Types

True/False Parameters

The true/false parameters in the configuration file are: `SEauto`, `SEfilling`, `SEjustify`, `SEdraw`, `SEtoken`, `SEignore`, `LPincludetag`, `LPincludecommands`, `LPdraw`, `LPcont`, `LPbegin`, and `LPtable`.

"True" or "False" may be spelled out completely or abbreviated, but only the first character is looked at. If it is a `T` or an `F`, the value is set appropriately, and the rest of the line is ignored. If no value exists, or if the first non-blank character is not a `T` or an `F`, the current value is not changed.

Numeric Parameters

The numeric parameters in the configuration file are: `SEleft`, `SEparagraph`, `SEright`, `LPnumber`, `LPfirst`, and `LPtop`.

The numeric values must have no spaces embedded within the number. The number is read starting with the first digit until the first non-digit is seen (e.g., space, letter, special character, or end-of-line). If no number is found, the current value is used.

Note that there is no `SEright` in the configuration file supplied with your system. If the value for `SEright` is not explicitly set in the configuration file the Editor will choose a value appropriate for the screen width of your computer.

One-Character Parameters

The one-character parameters in the configuration file are: `SEcommand`, `SEunderline`, and `SEescape`.

The command character *must* have a value, but the underline character and the Editor escape character need not have values. The command character is the first non-blank character after the parameter name `SEcommand`, if it's legal. If no legal value is found, the current value is used. For the other two, if a legal character occurs and there is no conflict, (see *Restrictions*, below), that is the character used. If an illegal value is found, the current value is used. If no non-blank character is found for `SEunderline` or `SEescape`, the parameter value is set to "none."

String Parameters

The string parameters in the configuration file are: `LPprinter`, `LPoutput`, `LPtag`, `LRheader`, `LRtopic`, and `LRpage`.

Letter case is always significant for string parameters. On `LPtag`, `LRheader`, `LRtopic`, and `LRpage`, the characters are taken exactly as specified, *including leading blanks*. Thus, you can specify whatever justification you want. Trailing spaces are ignored on all parameters except `LRpage`, so you can justify from the right margin in table of contents.

On `LPoutPut`, if you have the value `PRINTER:`, it must be *all uppercase*; leading spaces are removed from this parameter.

Combinatorial Parameter

The combinatorial parameter in the configuration file is: `LPdate`. This parameter specifies the format in which you want the date to be printed in the tag. The value of the parameter must be exactly four characters long. The first three characters are an `M`, a `D`, and a `Y`, in any order. There are six combinations: MDY, MYD, DMY, DYM, YMD, and YDM. The fourth character is the separator used between the parts of the date. For example, assuming the date is April 15, 1985, `MDY/` results in `4/15/85`, `DMY-` results in `15-4-85`, and so forth.

Saving Your Configuration File

When you Quit the Editor after creating or modifying a configuration file to suit your needs, Write to a file by the name of `EDITORC.ASC` *on the Editor's volume*. The "Editor's volume" is the volume the Editor resides on as defined in the *What* command in the Environment's main prompt. For example, if the Editor resides on volume `MySys:`, write the configuration file to `MySys:EDITORC.ASC`.

A configuration file need not have all these entries in it. If the values used when you do not have them specified in the configuration file are satisfactory, you do not need to include them in your configuration file.

Restrictions

The entries in the configuration file may be in any order; they do not need to be in the order specified above. However, one very important point in describing the configuration file is that *the parameter values are validated as they are encountered.* Thus, a single value may be considered invalid, and therefore be ignored, even if the next line would make it seem valid.

For example, suppose you want the command character to be a backslash (`\`) and the underline character to be tilde (`~`). If part of your configuration file was as follows:

```
SEcommand         \
SEunderline       ~
```

the command character would not be the backslash. The Editor sets the values to the defaults (those shown at the beginning of this section) *before* it reads the configuration file. The Editor ignores the line setting `SEcommand` to `\` because the underline character is currently a backslash (by default), and the command character must be different from the underline character. The next line, the `SEunderline` causes no conflict, so the underline character becomes the tilde.

To get what you want, just reverse the entries in the file:

```
SEunderline       ~
SEcommand         \
```

The Editor reads the `SEunderline ~` line, and changes the underline character accordingly. Then it reads the `SEcommand \` line, and since none of the other special characters is already the backslash, the command character becomes the backslash.

The same problem can occur in setting margins. If you want the left margin to be 80 and the right margin to be 90, the following file would not do it:

```
SEleft      80
SEright     90
```

The reason it would not work is this. The Editor first reads the `SEleft 80` line. The default right margin is 75, so when the Editor sees you want a left margin of 80, it thinks, "You can't have the left margin farther right than the right margin! I'll throw this one out." Again, though, you can get what you want by reversing the order of those two lines in the file:

```
SEright     90
SEleft      80
```

The Editor reads the `SEright 90` line, it checks and confirms that the requested right margin is greater than both the left margin and the paragraph margin. The value is less than 256, so it is therefore installed. Next, the Editor reads the `SEleft 80` line, confirms that the requested left margin is less than the right margin, and therefore installs it.

There are no other restrictions on the order of the entries in the configuration file.

You can confirm that your configuration file specifications worked by checking in the appropriate areas:

SE parameters	Go into the Editor, press S E.
LP parameters	Go into the Editor or the Lister and press L.
LR parameters	Generate a table of contents with the Lister.

Default Values for the Lister
The Lister gets its default values from the Editor configuration file, also. You must have the Editor volume as defined in the *What* command in the Environment's main prompt on-line when you enter the Lister in order for it to be able to use the values the configuration file specifies.

Productivity Hints

Chapter 6

This chapter describes several techniques you can use to increase your productivity. It has three sections. The first, *Miscellaneous Time Savers*, describes several things you can do to speed up your editing. The second section, *I Do the Same Thing Every Week*, discusses *streaming*, a technique that allows you to "automate" very repetitive tasks. The third section, *Working With Workfiles*, tells you about the workfile capability of the Editor, and includes a list of the Filer commands that relate to workfiles.

Miscellaneous Time Savers

The following techniques are time-saving practices to be aware of when using the HP TechWriter Editor.

Making Configuration File Access Faster

Whenever you enter the Editor, the configuration file is accessed at least once, and may be accessed twice. The first access is to determine if you have the correct codeword. This will always take place. At this point, the only thing the Editor cares about is the codeword, so it will search the configuration file until it either finds `CODEWORD`, or until the end of the file. Access time will be reduced if your codeword entry is (or entries are) the first thing in the file, as the sample in *Customizing the Default Environment* shows.

The second access of the configuration file takes place only if you start creating a new file, or if you load a file whose type is not `.TEXT`. In these cases, there is no environment associated with the file, so the configuration file is accessed to read your initial values for the Environment parameters.

The second access of the configuration file can be sped up by omitting definitions of parameters whose default values are satisfactory.

Speeding Up Searches

When the Editor is doing searches, the state of `Ignore case` is a significant factor in the speed of the process. Searches with `Ignore case` set to `False` are faster than searches with the value set to `True`, since case conversion is avoided during the search.

Searches with `Token search True` usually take more time than with it false. When doing a Token search, a Literal search is done first – each time a match is encountered the boundaries of the string are tested to see if it meets the definition of a token (remember, a string is a token only if it is bounded on the left and right by the start/end of the file, line endings, spaces, or special characters).

Speeding Up Margining

Two activities consume time during a margin operation: determining if a preceeding margin document command is still in effect (of the form `^^M,,ON`); and the line filling process of building a "paragraph". There are things you can do to reduce the total time it takes to margin your document.

Finding the Margin Values

Locking margins are used for each subsequent paragraph in the document until another margin document command (^^M) is reached. Thus, when you margin a paragraph, the Editor searches backwards in the file until it finds a margin document command and checks to see if it has the ON parameter. If there are no margin document commands, the Editor must search all the way to the beginning of the file before it can access the Environment parameters and begin building the paragraph.

To minimize this search time you should intersperse margin document commands every ten to fifteen screenfuls of text in a large document. This shortens the search distance in situations where you are not routinely using margin document commands. The margin document commands you insert for the sake of speed can be of the simple form ^^M (which means "use Environment values for all parameters").

Line Filling

As the Editor builds lines, underline characters and Editor escape sequences must be "ignored" from right margin calculations. Lines containing them will extend beyond the right margin on the CRT and align correctly when printed. In documents where you do not use underline or escape sequences, you should define the unnecessary Environment character to be blank. If the underline or Editor escape character is blank, the line scan for that character is omitted and noticeable time will be saved when margining a large number of paragraphs.

Useful Cursor Movements

Here are some common cursor movements which may come in handy. All of them assume the current direction is forward.

Moving the Cursor to the Left End of the Line

Press 0 P to move the cursor "zero" pages.

Moving the Cursor to the Right End of the Line

Press Return, then Backspace. (On the last line of the file press / ▶.)

Deleting the Rest of the Line

Press D to invoke the Delete command, then Return Backspace followed by Select. (On the last line of the file press D 9 9 9 ▶ followed by Select.)

Deleting the Entire Line On Which the Cursor Resides

First, move the cursor to the beginning of the line (0 P), then press D Return Select. (On the last line of the file press ▲ Return Backspace D, followed by 9 9 9 ▶ Select.)

Insert a Blank Line Above the Cursor Line

Move to the beginning of the line (0 P), and press I Return Select.

I Do the Same Thing Every Week!

Often there are routines that you go through regularly which are virtually, if not completely, identical. One example is preparing a memo to the rest of the staff of your company every Monday morning. The memo always has the same form, and goes to the same people, and only the date and the figures for the week have changed.

To save you time in doing repetitive tasks like this, there is a very useful HP TechWriter feature called *streaming.*

The concept of streaming involves putting a commonly-used sequence of commands into a file, and then telling the computer to perform the instructions in that file. The Stream command, which exists both in the HP TechWriter Environment and the Pascal operating system, tells the computer to get its instructions from the specified file, and, when the file is exhausted (or if an error occurs), to return control to the keyboard for subsequent input.

Making and Streaming Stream Files

To build a stream file to perform a particular task, you must know the sequence of commands needed to perform that task. You may want to execute the operations by hand once, while noting everything you type in response to the computer's prompts. Note particularly the occurrences or absences of the (Return) key. If you encounter a question that asks a `(Y/N)` or `(R/O/N)` question do not record any answer for use in the stream file. These kinds of questions will be answered automatically as the file is streamed. `(Y/N)` (Yes/No) questions will always be answered "`Y`", and `(R/O/N)` (Remove/Overwrite/Neither) questions will always be answered "`R`".

Once you know the command sequence for your task, use the Editor to enter the keystrokes you noted into a file. To achieve the effect of pressing the (Return) key from the keyboard, start a new line in the stream file.

Once you have built the stream file, you can stream it by pressing (S) from the Environment menu, and then typing the name of the file. Note that, as in the Editor, `.TEXT` will be automatically appended to the end of the file name unless you end it with a period. As the file is streamed, the characters are consumed by the operating system. After all the characters in a stream file have been interpreted, control is returned to the keyboard.

A Stream File Example

You can create a stream file to specify locations of files, peRmload certain files, specify the default volume, etc. These are things which you might want to do every time you turn your computer on. You could do the entire sequence of things by hand each time, or create a stream file which does it. Then, all you need to do is tell the system to stream the file, and all the individual commands will be executed. Needless to say, this can save you much typing.

An example stream file is shown below. The commands used in this example are discussed in detail in *The HP TechWriter Environment* chapter.

```
*  peRmload Editor and Filer                  (1)
rerf                                          (2)
*  Set default volume to "Myvol:"             (3)
wdMyvol:                                      (4)
q                                             (5)
```

The numbers on the right **are not** part of the stream file; they are merely for reference in the discussion. As usual, command characters may be upper or lower case.

The lines with asterisks as their first characters (1 and 3) are comment lines. As such, they all could be removed without affecting the operation of the stream file at all. A line which has an asterisk as the first character is considered a comment **only if it is encountered while at the main command level of the Environment**. If a comment is not at the main command level, the asterisk and the comment itself will be interpreted as characters to be processed. Comments cannot be embedded among commands for subsystems or user programs. When the command interpreter encounters one or more comment lines while streaming, the comments are displayed briefly on the screen, thus allowing the process to be monitored.

Line 2 peRmloads the Editor and the Filer. The "r" invokes the peRmload command, which then prompts for the program to load. "e" is for Editor; "f" is for Filer, etc. See the *peRmload* command.

Lines 4 and 5 define a new default volume. This is the volume name that will be used to access files specified without any volume name. Line 4 invokes the *Whatfiles* command (w) and specifies that the default volume (d) is "Myvol:".

An attempt to place a comment between lines 4 and 5 would have caused problems. A "comment" at that location, since it was in the *Whatfiles* menu and not the main command level, would have been interpreted as characters to be processed. A comment is only allowed at the main command level.

The "q" in line 5 quits the *Whatfiles* menu. The "q" could not have been at the end of line 4, which specified the default volume, because the volume name needs to be terminated with a Return. In a stream file, a Return is achieved by an end-of-line.

Control Characters in Stream Files

Stream files can be used to run programs as well as to change the HP TechWriter Environment. In the Editor, it is sometimes necessary to use keys that also act as immediate commands, such as Select or Backspace. Substitute the following key sequences for those keys.

Key	**Substitute**
▲	CTRL-Select ?
▼	CTRL-Select J
▶	CTRL-Select <
◀	CTRL-Select H
Backspace	CTRL-Select H
Select	CTRL-Select C
Shift-Select	CTRL-Select ;
Tab	CTRL-Select I
Clear display	CTRL-Select L
Return	CTRL-Select M
Bell	CTRL-Select G

The CTRL-Select keypress is displayed on the screen as an inverse-video "K", thus: K. This character has an ASCII value of 255. It can also be generated by pressing ANYCHAR 2 5 5. (To access the ANYCHAR function from an HP 46020A keyboard, press SHIFT-f5.[1]) When the computer encounters a K while streaming, it is interpreted as a signal that says, "There is a non-ASCII keystroke coming up." Then the computer looks at the following character to see *which* non-ASCII character you wanted. If the K is the last character in a line, the computer ignores it, and the end-of-line is interpreted as an Return keypress.

Any control character can be caused through a stream file. The control character (ASCII codes 0 through 31) which is placed into the keyboard buffer from the stream file is defined as: "The ASCII code (of the character following the K in the stream file) modulo 32." "Modulo" means "remainder." For example, 47 modulo 32 is 15 because there is 15 remaining after you take all the 32s you can out of 47.

Suppose you want to place a shift-out character into the keyboard buffer (we can't imagine why you would do this). The shift-out character has an ASCII code of 14. Any of the three sequences K., KN, or Kn would result in a shift-out character being placed into the keyboard buffer. Any of the three could be used because the ASCII codes of ".", "N", and "n" are 46, 78, and 110, respectively. Those three numbers differ by 32s, so all three numbers modulo 32 result in 14, the ASCII code for the shift-out. For the rest of the ASCII characters, see the ASCII table in the appendix at the end of this manual.

Asking Questions from a Stream File

A stream file can be made to display a prompt on the CRT and then wait for an input string from the keyboard. The input string is assigned to a variable in the stream file. When the variable is encountered during streaming, the string is used in its place.

Input prompts must appear in the stream file before any of the commands. An input prompt is denoted with an equals sign (=) as the first character on a line.

To prompt for an input string, place an equals sign, followed by a single alphanumeric character variable (upper and lower case letters used for variables are considered identical), followed by the prompt text on the line. Up to 36 prompts are allowed. Their names may be "A" through "Z" and "0" through "9". For example:

```
=fWhat is the name of the file to be edited ?
```

When the file name is typed in response to the prompt during stream file execution, it will be stored in the specified variable, in this case the variable named f.

After typing all the prompts, begin entering the commands in the stream file. When you want the input string to be given to the operating system, use the prompt variable preceded by "@". For example:

```
e@f
```

is a command stream that invokes the HP TechWriter Editor. The file whose name was given in response to the above prompt is then loaded.

1 For further information on the softkeys for the HP46020A keyboard, see the *Pascal 3.0 User's Guide* in the section entitled *HP46020A Keyboard.*

Autostart Files

Another example of a stream file is called an "autostart" file. As its name implies, an autostart file automatically starts a process or sequence of commands. An autostart file is nothing more than a stream file which has the name `AUTOSTART` (all capital letters, and no suffix), and which resides on the volume which is the system volume at bootup time.

An autostart file is always sought during the boot process. If a file named `AUTOSTART` exists on the system volume, it is automatically streamed. If you do not wish a file to be streamed automatically, merely make sure there is not a file called `AUTOSTART` on the power-up system volume.

You could modify the stream file example given earlier to make it an `AUTOSTART` file by naming it `AUTOSTART`, placing it on a volume which is the system volume at bootup, and adding the lines shown below.

```
<blank line>                                  (1)
<blank line>                                  (2)
*  peRmload Editor and Filer                  (3)
rerf                                          (4)
*  Set default volume to "Myvol:"             (5)
wdMyvol:                                      (6)
q                                             (7)
*  Invoke the time/date screen                (8)
t                                             (9)
```

The two blank lines at the beginning are used to respond to the date and time prompts that the system displays when it starts. The "t" in line 9 invokes the time command again to give you another chance to enter the date and time.

An autostart file streams by itself, but you can still stream it by hand at any time by pressing `S` from the HP TechWriter Environment's main command level, and specifying the name of the file: `AUTOSTART.`. The period at the end of the file name is important. When streaming, the name of the file will have `.TEXT` appended to it unless you place a period at the end of the file name.

If you are creating an autostart file by exiting the Editor using the Write option and specifying `AUTOSTART` as the file name, be sure you append a period to the name `AUTOSTART`. When specifying a file name to the Editor, the name of the file will have `.TEXT` appended to it unless you place a period at the end of the file name. The autostart file's name must be `AUTOSTART`, not `AUTOSTART.TEXT`.

Streaming Errors

The streaming process creates a temporary file called STREAM on the system volume for use during the streaming process. Thus, if there is no more room in the volume for the temporary file, you will get an error similar to the following:

```
Can't create *STREAM

-------------------------------------------------
error -10: tried to read or write past eof
PC value:     -14500
```

The PC (program counter) value will probably be different, but the cause of the problem will be the same.

Another potential cause of a streaming problem is a directory overflow. If the directory on the system volume is full, you will get a message similar to the following:

```
Can't create *STREAM

-------------------------------------------------
error -10: no room in directory
PC value:     -14500
```

Working With Workfiles

A workfile in the Editor is used as a "scratchpad" version of a text file. If a workfile is present, it will automatically be read in when you start the Editor. If you are repeatedly working with one file, this means that you don't have to keep typing its name. However, while you have a workfile present, the Editor will not give you the opportunity to enter the name of any other file before reading the workfile.

Exiting the Editor (using the Quit command) gives you the option of Updating the workfile. If the Editor was entered with a workfile (or if the Update option was used earlier in the same editing session), the Editor writes the contents of the text file in memory to the file called `*WORK.TEXT` (the `*` is a shortcut method for specifying that the file is to go onto the system volume). When you are through with all your editing, it is a good idea to enter the Filer subsystem and Save the workfile.

If you have been using a workfile but now want to edit a different file, you *must* enter the Filer and use the `New` command, which purges the workfile.

The Filer commands which deal with the workfile are `Get`, `Save`, `What`, and `New`.

Task Reference

Chapter
7

This chapter describes the commands you need for common tasks. For more information, see the command entry in the Reference Chapter, or the indicated section.

Adding Text	*Insert* in the Editor.
Aligning Lines of Text	*Adjust* in the Editor.
Archiving Your File	*Quit* in the Editor or Lister.
Boldface Printing	^^*S n=0* document command in the Editor. See also *Printer Escape Sequences.*
Centering Text	*Adjust Center* in the Editor.
Changing the Name of a Disc Directory	*Change* in the Filer.
Changing the Name of a File	*Change* in the Filer.
Changing the Page Number	^^*P N=* document command in the Editor.
Conditional Page Break	^^*P C=* document command in the Editor.
Consolidating Disc Space	*Krunch* in the Filer.
Copying Files into Text	*Copy File* in the Editor.
Copying Files on Discs	*Filecopy* in the Filer.
Counting Occurrences of a String	*repeat factor Find Literal* in the Editor.
Counting Occurrences of a Word	*repeat factor Find Token* in the Editor.
Creating Tables	*Set Program mode* and precede with ^^*M off* document command; in the Editor.
Date in Footers	*<date>* token in text line following ^^*T* document command in the Editor.
Disc Directory Contents	*List* or *Extended directory* in the Filer.
Disc Volume List	*Volumes* in the Filer.
Duplicating Text	*Delete* text, then *Copy Buffer* in the Editor.
Deleting Files	*Remove* in the Filer.
Deleting Text	*Delete* or *Zap* in the Editor.
Disc Initialization	Use the Mediainit program. See *Disc Initialization* in the *Protecting Yourself* chapter.
Finding Strings	*Find Literal* in the Editor.

Finding Words	*Find Token* in the Editor.
Fonts	Use escape sequences to access the special fonts of your printer. See the section *Printer Escape Sequences*; *Set Environment*, *Escape ch* in the Editor.
Footers	*^^T* document command in the Editor.
Forms	See the section *Designing Forms*.
Help	*Help* in the Editor, Lister, or HP TechWriter Environment.
Ignoring Upper/Lower Case	*Set Environment Ignore case True* in the Editor.
Including Other Files	*Copy File* in the Editor
Initializing Discs/Media	Use the Mediainit program. See *Disc Initialization* in the *Protecting Yourself* chapter.
Justifying Text	Use the *Margin* command, the ^^Margin document command, or the *Adjust* command.
Listing Contents of a Disc Directory	*List* or *Extended directory* in the Filer.
Listing Disc Volumes	*Volumes* in the Filer.
Listing (Printing) File Contents	*List* command in the Editor or Lister.
Listing (Printing) Partial File Contents	*^^L off* document command to turn off listing of section of a file, then *List* command in the Editor or Lister.
Listing (Printing) Multiple Files' Contents	Use the Lister program.
Left-Justifying Text	*Adjust Left* in the Editor. See also Margin.
Local Margins	*^^M* document command in the Editor.
Margining Text	*Set Document* mode, then *Margin* in the Editor.
Modifying Text	*Xchange* or *Replace* in the Editor.
Media Initialization	Use the Mediainit program. See *Disc Initialization* in the *Protecting Yourself* chapter.
Moving the Cursor to the Beginning of the File	*Jump Begin* in the Editor.
Moving the Cursor to the End of the File	*Jump End* in the Editor.
Moving the Cursor from Where You Are	Use the space bar, the (Return) key, the (Tab) key, the (Backspace) key, the arrow keys ((◀), (▶), (▲), and (▼)), or the *Page* command in the Editor.
Moving the Cursor to a Specified Point	*Jump* to a *Marker*, previously specified with the *Set Marker* command.
Moving Text	After a deletion with the *Delete* or *Zap* commands, move the cursor to the new position, then *Copy Buffer*.

Multiple Copies of Output	When in the List menu, queue up *n* keystrokes of `L` in the type-ahead buffer. See the *List* command in the Editor or the Lister.
Outdenting Paragraphs	*Set Environment, Para margin* less than *Left margin* in the Editor.
Overriding Margins	*^^M* document command in the Editor.
Packing Files on a Disc	*Krunch* in the Filer.
Page Breaks (New Page)	*^^P* document command in the Editor.
Page Numbers in Footers	*<page>* token in text line following *^^T* document command in the Editor.
Pictures in Your Document	Create a picture with a graphics program (or your own Pascal or BASIC 3.0 program), plot it to a file, run the Picture Processor on it, and use a *^^D* drawing command in your document.
Printer Escape Sequences	*Set Environment, Escape ch* in the Editor. Use escape sequences to access the special capabilities of your printer. See the section *Printer Escape Sequences.*
Printing a File	*List* command in the Editor or Lister.
Printing Part of a File	*^^L off* document command to turn off listing of section of a file, then *List* command in the Editor or Lister.
Printing Multiple Files	Use the Lister program.
Protecting Text "As-Is"	*^^M off* or *^^M off on* document commands in the Editor.
Ragged Left Margins	*Adjust Right* in the Editor.
Ragged Right Margins	*Set Environment, Justify False* or use *^^M J=F* document command in the Editor.
Recovering Text	*Copy Buffer* in the Editor.
Redrawing the Screen	*Verify* in the Editor.
Remargining the Whole Document	`/` *Margin* in the Editor.
Removing Files	*Remove* in the Filer.
Removing Text	*Delete* or *Zap* in the Editor.
Renaming a Disc Directory	*Change* in the Filer.
Renaming a File	*Change* in the Filer.
Re-Painting the Screen	*Verify* in the Editor.
Right-Justifying Text	*Adjust Right* in the Editor.
Saving a Workfile	*Save* in the Filer.
Saving Your File	*Quit* in the Editor or Lister.

Setting the Cursor Direction Backward	Press $\boxed{,}$, $\boxed{<}$, or $\boxed{-}$ in the Editor.
Setting the Cursor Direction Forward	Press $\boxed{.}$, $\boxed{>}$, or $\boxed{+}$ in the Editor.
Setting the Default Volume	*Prefix* in the Filer or *What Default* in the Environment.
Setting the Prefix	*Prefix* in the Filer or *What Default* in the Environment.
Setting Up a Specified Point	*Set Marker* in the Editor.
Side-by-Side Text and Graphics	Include text between *^^D* drawing command and its terminator. Also, see section *Side-by-Side Text and Graphics.*
Smooth Left Margins	*Set Document mode* in the Editor.
Smooth Right Margins	*Set Environment, Justify True* or use *^^M J=T* document command in the Editor.
Superscripts/Subscripts	Use escape sequences to access the special capabilities of your printer. Either use half-linefeeds or use a math font. See the section *Printer Escape Sequences*; *Set Environment, Escape ch* in the Editor.
Table of Contents	Use *^^C* document commands in the Editor to mark entries, then set *List* command *tAble of contents True* and list file(s) in the Lister.
Tables	*Set Program mode* and precede with *^^M off* document command; in the Editor.
Time in Footers	*<time>* token in text line following *^^T* document command in the Editor.
Underlining Text	*Set Environment, Underline ch,* then surround text to be underlined with the underline character; in the Editor.
User-Defined Characters	Overprint characters *^^S n=0.* See the section *User-Defined Characters.*
Word Wrap	*Set Document mode* for *Insert* and *Margin* commands; in the Editor.

The HP TechWriter Environment

Chapter 8

Introduction

The HP TechWriter programs are: the *Editor*, the *Lister*, the *Picture Processor*, the *Filer*, and *Mediainit*. These programs can only be run from an *environment* program. An environment program simply provides information to all of the programs about what kinds of discs and peripherals you have and where they are.

The HP TechWriter programs may be run from one of two environments. If you own the Pascal 3.0 operating system, you can run the HP TechWriter programs from that environment if you wish to do so. Otherwise, you must run the HP TechWriter programs using the environment delivered on the `BOOT:` disc.

This chapter relates to commands available from the HP TechWriter Environment only. If you are using the Pascal operating system, this chapter does not pertain to you. Instead, you should read the *Pascal 3.0 User's Guide* to learn about the commands available for your operating system.

Environment Programs

The HP TechWriter Environment has two prompt lines. The first prompt line looks like this:

```
TWenv:  Editor  Picprocess  Lister  Filer  Mediainit  Graphics  ?
```

These are the programs available with HP TechWriter and a convenient mechanism to include access to a graphics editor program of your own choosing (it must be a Pascal 3.0 code file). Pressing the capitalized letter of each program's name causes that program to be loaded and executed.

Pressing `?` causes the prompt line not currently visible to be displayed. The second prompt line will be discussed in the *Environment Commands* section later in this chapter.

Descriptions of the programs accessed by the Editor, Picprocess, Lister, and Mediainit commands have already been presented in the previous chapters.

Filer

The Filer provides several commands for maintaining volumes, files, and workfiles. Filer commands list volumes on-line, list the directory of a volume, scan discs for bad spots, collect disc free space, and create new directories on volumes. Using the Filer, you can create, copy, rename, and remove files or convert files from one type to another. Other Filer commands identify, save, and clear workfiles.

You used the Filer in earlier chapters of this manual. This section contains a brief introduction to the Filer. The *Filer Reference* chapter contains a more complete description of each Filer command. Please consult the *Pascal 3.0 User's Guide* if you want more information about files, volumes, workfiles, and the Filer.

Entering the Filer

To enter the Filer, insert the disc labeled `ACCESS:` and press F. The screen clears and displays the Filer prompt on the top line:

```
Filer: Change Get Ldir New Quit Remove Save Translate Vols What Access Udir ?
```

You are now in the HP TechWriter Filer subsystem. The Filer prompt shows the most common commands used in the Filer and "prompts" you to give the subsystem a command.

The prompt shows only a partial list of the available commands. To see the others, type ?. The prompt line shows the Filer's alternate prompt:

```
Filer: Bad-secs Ext-dir Krunch Make Prefix-vol Filecopy Duplicate Zero ? [3.0]
```

The alternate prompt displays the revision number of the Filer in brackets. Type ? again and the main Filer prompt reappears.

Filer commands are initiated by typing a single key corresponding to the first character of the command shown in the Filer prompt.

All of the commands in the Filer operate in one of two ways. The Filer either performs the operation immediately when you initiate the command or requests a volume specification or a file specification before the operation begins.

Filer operations can be aborted by typing Shift-Select when a single character is expected and Shift-Select Return in place of a file specification.

Volume Specifications

A volume specification identifies a volume. You can supply either the name of the volume or its associated unit number, followed by a colon (:). To specify the *default volume* you can use a colon (:) alone. To specify the system volume you can use just an asterisk (*). We'll learn more about default and system volumes later in this section.

To find out what volumes are available, you can use the *Vols* command to list available volumes.

Listing Volumes

Type (V) for the `Vols` command. The Filer displays a list similar to this:

```
Volumes on-line:
  1   CONSOLE:
  2   SYSTERM:
  3 * ACCESS:
  4 # EDIT:
  6   PRINTER:
Prefix is - EDIT:
```

The number in the first column is the *unit number* for the volume. You may use either the unit number or volume name to specify volumes to HP TechWriter subsystems.

A * in the second column identifies the *system volume*. At power-up, the system designates the highest performance mass storage device as the system volume and then uses a small portion of this volume for special system functions. The system volume remains fixed unless you use the `What` command to change it (see the *Whatfiles* section later in this chapter).

A # in the second column identifies a *block structured* device, normally a disc, which can be used to store and retrieve files.

The third column lists the volume name.

The last line of the Volume listing identifies the *prefix*, the name of the *default volume.*

Default Volume

When you frequently access several files on the same volume, it is tedious to repeatedly type the volume name or unit number. If you omit the volume name from a file specification, the HP TechWriter system will look for the file on the *default volume.*

Use the `Prefix` command to specify the default volume. Type (P) and then the name of the new default volume. You can see what the current default volume is by pressing (P) (Return).

Bad Blocks on Volumes

The flexible discs you use for HP TechWriter volumes may become unreadable after excessive use or abuse. The Filer's `Bad-secs` command allows you to check flexible discs to find out if each block (sector) is readable.

To initiate the `Bad-secs` command press `B` and answer the prompt with the volume specifier of the disc you want to scan for bad blocks. The Filer displays a message indicating that it is scanning the volume from block 0 to the end of the volume. The Filer displays the sector number of any bad sector it finds.

A file with a bad sector is usually unreadable. After finding a bad sector you can use the Filer's `Ext-dir` (Extended directory) command to find which file is written on the portion of disc containing the bad sector. Use the `Change` command to change the file type to `.BAD`. If the bad sector is in an unused area of the disc, you should use the `Make` command to create a dummy file over the bad sector.

Remember that flexible discs *DO WEAR OUT*. You should frequently make backup copies of your discs to protect important files. A single bad sector indicates that your disc is wearing out and should be replaced. The disc should only be used if you are willing to risk losing information on that volume.

No Room on Volume

Obviously, there is a limited amount of space in a disc volume. When there is not enough room on a volume to create a new file, the system will report an I/O (input/output) error.

As files are created and removed, the disc will develop "holes" of free space between files. This is called "fragmentation." It is possible for a considerable amount of free space to exist in the volume, yet be unuseable because the free space is in pieces too small to use.

You may be able to solve this problem using the Filer's `Krunch` command. This command consolidates all of the volume's free space by moving all of the files on a volume to the front of the volume. Files with `.BAD` file type will not be moved in a Krunch operation. To see how fragmented your volume is, use the Filer's `Ext-dir` List command. This command lists both the files the fragmented space on the volume.

Listing a Directory

To find out what files are on a volume, type `L` to initiate the `Ldir` (List directory) command. The Filer prompts you to specify the volume whose directory is to be listed and then lists information about files in the directory.

Detailed Directory Listing

To get a more detailed disc directory listing, press `E` to invoke the `Ext-Dir` (Extended Directory) command. The Filer will prompt for a volume name as in the `Ldir` command.

The Extended directory listing contains the same information as the List directory listing plus the number of the block where the file starts, the type as recognized by the file system, the type-code used by the directory system, SRM access information and other information.

File Specification

A file specification includes both a volume specification and a file name. If the volume specification is omitted and only the file name is given, the Filer looks for a file of that name on the default volume.

You should choose file names consisting of alphabetic letters and digits. Only the hyphen, underscore, and period punctuation marks are acceptable within a file name. Blanks are removed from file names.

File names are limited to nine characters not counting the file suffix.

Renaming Files and Volumes

The `Change` command allows you to rename files and volumes. This command requires two specifications: the original name and the new name. Press `C` to invoke the `Change` command.

Removing Files

Use the `Remove` command to delete files from a directory. Press `R` to initiate the `Remove` command and type the name of the file to be removed.

Moving Files from Volume to Volume

The `Filecopy` command allows you to copy files from one volume to another or to a different place on the same volume. Type `F` to invoke the `Filecopy` command, then enter the name of the file to be copied and the name of the destination file.

If the destination file already exists the Filer will ask you to decide whether to remove the old file or cancel the Filecopy operation.

Copying Entire Volumes: Backup Copies

In the *Protecting Yourself* chapter we described a backup process suitable for copying all files on a volume to another volume.

Using HP TechWriter Files in the BASIC Language System

You can use HP TechWriter to create files for BASIC Language System programs, and you can use HP TechWriter to edit files created by BASIC programs. But there are file name and file type limitations that you must follow to share files between HP TechWriter and BASIC.

The HP TechWriter file system allows you to use most ASCII characters in a file name. But to share files with the BASIC Language System you must restrict file names to only upper-case letters, digits. and the underscore character, and file names cannot contain more than nine characters.

The nine-character file name length would be a very severe restriction when four or five characters are required for a suffix. To ease this problem, the HP TechWriter system automatically compresses the suffix. Normally, you never see this suffix compression because the change is automatically reversed in directory listings. But if you examine a directory using the BASIC Language System, the file name will be different.

Here is how HP TechWriter compresses the file name before putting it into the directory. If a standard suffix is found, (for example, TEXT or ASC) it is removed from the name and the dot (.) is replaced by the first character of the suffix. If the resulting name is longer than ten characters, an error is generated. If the new name has fewer than ten characters, it is extended to ten characters by appending underscores (_). If the file name does not have a standard suffix, the file is a data file and the name is used unchanged unless it is too long. If it is longer than ten characters, an error is reported.

The following examples illustrate suffix compression:

File name	Compressed name
'A.ASC'	'AA________'
'charlie'	'charlie '
'123456789.TEXT'	'123456789T'
'MyHugeTEXT'	rejected because it would be confused with 'MyHugeTEX.TEXT'

Files created and edited by the HP TechWriter Editor normally have the .TEXT suffix. To use a HP TechWriter file in the BASIC Language System you must use the Translate command to convert the .TEXT file to .ASC.

Translating Files

The HP TechWriter system supports several different types of "text" files. These files are usually created by the Editor and can be programs, documents, or data. When the file is stored on a disc, the internal format of the file is determined by the suffix appended to the file specifier. The different formats have different information in the file header and can have different end-of-line schemes.

You can use the Translate command to convert from one file type to another. The various file formats recognized by the HP TechWriter system are TEXT, ASCII, and DATA. No suffix indicates a DATA file, a .TEXT suffix indicates a TEXT file and a .ASC suffix indicates an ASCII file.

Press T to invoke the Translate command. The Filer will prompt for the name of the file to be translated and the output file.

The Filer will create an output file of a type corresponding to the suffix on the output file name, read the text data from the input file, reformat the data to match the output file type, and write the data to the output file.

Translate is used to convert files from .TEXT. to .ASC when HP TechWriter files are to be accessed from the BASIC Language System.

Listing Files to the Printer and Screen

The Translate command is also used to send files to the printer or to the screen. Logically, the printer and screen are just files of a different format.

To send files to the printer or screen press T and then type the file name. The output file would be PRINTER: for the printer connected to your computer or CONSOLE: for your computer's display screen.

The System Workfile

The HP TechWriter features a workfile, which can be used by the Editor and Filer.

If the text version of a workfile exists when you enter the Editor, the Editor never prompts you for a name of the text file to edit but reads the workfile instead.

The Filer `Get` command is used to specify a workfile. The `Save` command allows you to save the workfile. The `New` command is used to release the workfile. The `What` command is used to find out information about the current workfiles.

Filer Commands for SRM Systems

The `Access`, `Udir` and, `Duplicate` commands are used only if you have a Shared Resource Management (SRM) system. Consult the *Pascal 3.0 Workstation System* manual for more information on SRM systems.

Leaving the Filer

Exit the Filer by pressing (Q) for `Quit` from the main Filer prompt. You will immediately be returned to the HP TechWriter Environment.

You can also exit the Filer by pressing the (Stop) key. The (Stop) key waits for any current disc I/O to complete before it actually executes. This key can be used at any time – even while executing a Filer command. However this practice is not recommended since some commands may cause damage to your files if (Stop) is pressed while the files are being accessed.

Environment Commands

This section discusses the commands shown on the second HP TechWriter Environment prompt line. The second prompt line looks like this:

```
TWenv:  Time  Whatfiles  peRmload  Stream  Help  ?
```

Time

This reruns the part of the boot-up sequence which allows you to set the system clock's date and time. After you press T, the screen is erased, and a prompt similar to the following appears:

```
New system date ?

  System date is          1-Mar-00
  Clock time is           00:00:23

  HP TechWriter           [Rev. 1.0   1-Jul-84]

  Total Available Memory 100000 bytes


  System  volume:   ACCESS:
  Default volume:   ACCESS:


Copyright 1983 Hewlett-Packard Company.
All rights are reserved.  Copying or other
reproduction of this program except for archival
purposes is prohibited without the prior
written consent of Hewlett-Packard Company.
```

Enter the current date in the form shown: day, month, year. For example, March 14, 1985 is input as `14-Mar-85` Return. After you answer this question, the value just entered is placed in the `System date is` area, and the following prompt for the time comes up:

```
New system clock time ?
```

Enter the current time in the form: hour, minute, second. The time is in a 24-hour format. For example, 2:15:04 p.m. would be entered as `14:15:04` Return.

When you specify a date and time, the new values are placed into the directory of the system volume. This date is retrieved the next time the Time command is executed during the boot-up process. Thus, you know the last time the system was used.

You need only specify as much of the date as has changed since the displayed date. For example, if the date displayed is August 7, 1984, and you turn the machine on on August 8, 1984, you need only specify the new day, since the month and the year are still the same.

The values you specify for the time are taken as is, and the values you don't specify are considered zero. For example, `8:4` is interpreted 8:04:00 a.m., and 14 is considered 2:00:00 p.m.

The values entered here will remain current inside the computer as time goes on, and the current date and time will be written onto the directories of discs as you create or update your files. Thus, you can always determine when the last modification was made to a particular file.

The date and time questions need not be answered if you do not wish to do so. If you just press Return in response to the questions, their defaults will be March 1, 1900, and 12:00 midnight. Both the time and the date are continually updated from there.

After these two questions have been answered, the main HP TechWriter Environment prompt returns.

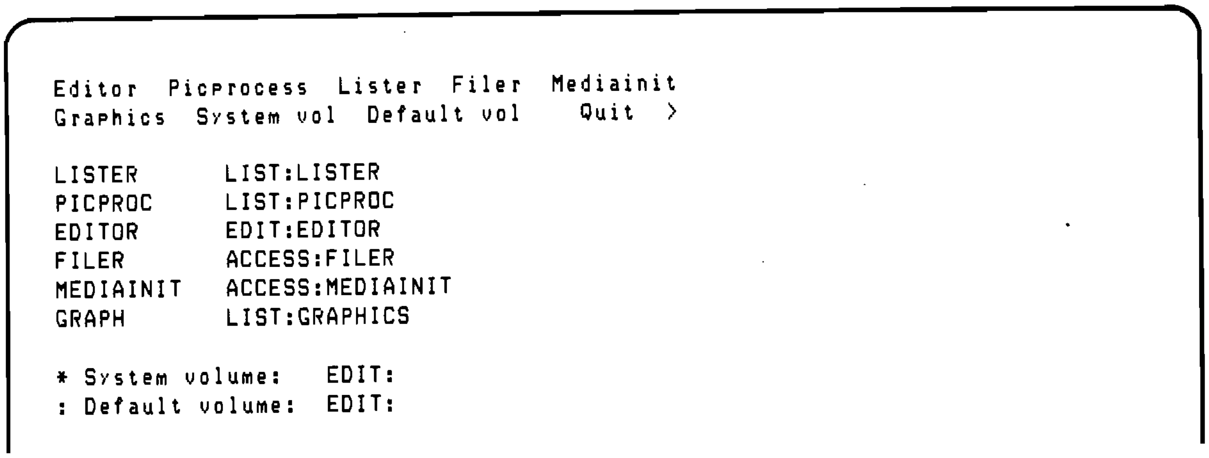
Whatfiles

The `Whatfiles` command lets you specify the default and system volumes. The default volume is the place where all programs look for your files if you do not specify a volume name. The system volume is the volume on which the workfile resides, and on which the temporary file is created during streaming.

The `Whatfiles` command also lets you specify where the major programs in HP TechWriter reside. The major programs in the HP TechWriter Environment are the Editor, the Lister, the Picture Processor, the Filer, and Mediainit. The HP TechWriter Environment also has room for you to add a graphics editor should you wish to do so. The `Whatfiles` command makes it possible for the Environment to find the HP TechWriter programs if you move them to other mass storage devices to take advantage of greater speed or reliability than the floppy discs afford.

Say, for example, you have moved them to a hard disc volume. You may now need to tell the computer that when you press E for the Editor, L for the Lister, P for the Picture Processor, etc., you want it to look for them on the hard disc rather than the floppy disc on which they were delivered. The Environment program automatically looks for the programs when it is booted. You can use the `Whatfiles` command to make sure it found them, and to tell it where they are if it did not.

When you press W from the main command level of the HP TechWriter Environment, a menu similar to the following comes up.



```
Editor  Picprocess  Lister  Filer  Mediainit
Graphics  System vol  Default vol    Quit  >

LISTER        LIST:LISTER
PICPROC       LIST:PICPROC
EDITOR        EDIT:EDITOR
FILER         ACCESS:FILER
MEDIAINIT     ACCESS:MEDIAINIT
GRAPH         LIST:GRAPHICS

* System volume:   EDIT:
: Default volume:  EDIT:
```

The file specifiers to the right of the words LISTER, PICPROC, EDITOR, etc. are the files which will be loaded when you press the appropriate letter from the main command level of the HP TechWriter Environment. The system wakes up thinking that the HP TechWriter Editor is named `EDITOR` and it resides on the volume `EDIT:`. Indeed, this is the file name and location for the Editor as it comes from Hewlett-Packard. If you wish to change the file name and/or location, you must tell the Whatfiles command, so the system can load the correct file on demand.

For example, say you want the Editor to be called `TWEditor` and you wish to place it on volume `SYSTEM:`. From the main command level of the HP TechWriter Environment, you would press W to enter the Whatfiles menu. Then press E to indicate that you wish to change the Editor's definition. The current Editor file name and volume name disappear, and the cursor is placed there, ready for input. If you decide you do not want to change the definition, just press Return after having typed nothing (or having erased everything you did type). To make the change, type `SYSTEM:TWEditor` Return. The new name and location of the file containing the Editor are now installed.

You can do similar things to any of the other program names.

Specifying the System Volume
You can redefine the system volume by pressing S followed by a volume name (do **not** include a file name here) and Return.

When you press S from the Whatfiles menu, the current system volume name disappears and the cursor is placed there, ready for input. If you decide you do not want to change the definition, just press Return after having typed nothing (or having erased everything you did type). If you do want to change it, you reply by giving the volume name or the logical unit number of the volume you wish to be the new system volume. Terminate the entry by pressing Return. After the system looks at all volumes which are currently on-line, and assuming you specified a valid volume, the requested volume will become the new system volume.

Note

When you change the system volume, the on-line volumes are scanned beginning with the new system volume, to find the six HP TechWriter programs. The What table is updated to reflect the volume name where each program is found. For each program not found, the volume name is set to the default (shown above). You should specify the new system volume **before** modifying the volumes and/or file names of the programs.

Specifying the Default Volume

You can also redefine the Default volume; that is, the volume on which files are sought unless a volume is explicitly specified.

When you press D from the Whatfiles menu, the current default volume name disappears and the cursor is placed there, ready for input. If you decide you do not want to change the definition, just press Return after having typed nothing (or having erased everything you did type). If you do want to change it, you reply by giving the volume name or the logical unit number of the volume you wish to be the new default volume. Terminate the entry by pressing Return. Assuming you specified a valid volume, the requested volume will become the new default volume.

Leaving the Whatfiles Command

When you press Q, the main Environment prompt returns.

peRmload

When going back and forth between two or more programs, waiting for them to be loaded every time you want to use them can get annoying. Two programs which are typically involved in this process are the Editor and the Filer. To avoid the wait, there is an option to *permanently load* a program into memory. This option is called *peRmload* in the HP TechWriter Environment.

You can use this command to permanently load any of the HP TechWriter programs into memory. Doing this makes program access very fast by eliminating the need to load the program from disc whenever it is executed after quitting a different program (that is, repeated executions of the *same* program do not require repeated load operations).

However, when you *peRmload* a program it remains in memory until you reboot the system. This means that the program takes up memory space whether you are using it at the moment or not. If you have restrictive memory constraints, peRmloading may not be a viable alternative for you.

Permanent Loading From the HP TechWriter Environment

When you press R from the main command level of the HP TechWriter Environment, the following prompt is displayed:

```
ENTER LETTER OF PROGRAM TO LOAD:  >
Editor  Picproc  Lister  Filer  Mediainit  Graph
```

The second line of the prompt ("Editor Picproc ...") shows the same list of programs available in HP TechWriter as is shown in the Whatfiles command. Press the first letter in the name of the program you wish to load. The appropriate program as currently set in the Whatfiles program list will be loaded.

Stream

The Stream command directs the computer to get its instructions from the specified file, and, when the file is exhausted (or if an error occurs), to return control to the keyboard for subsequent input.

The concept of streaming a file involves putting a commonly-used sequence of commands into a file, and then telling the computer to perform the instructions in that file.

After you press S the following prompt line will appear:

```
Stream what file ?
```

Type the name of the file containing the commands you want executed, followed by Return. Notice that, as in the Editor, .TEXT will be automatically appended to the end of the file name unless you end it with a period. As the file is streamed, the characters are consumed by HP TechWriter. After all the characters in a stream file have been processed, control is returned to the keyboard.

Instructions for building the contents of a stream file, and the kind of errors that can occur while streaming a file, are presented in the *I Do the Same Thing Every Week!* section of the *Productivity Hints* chapter.

Help

Once you are familiar with the HP TechWriter Environment, you may find the information displayed by pressing H to be helpful in reminding you of a particular command's operation.

Press any key to clear the help display from the screen, prior to pressing the next command.

Editor Reference

Chapter 9

Editor Commands

This chapter contains a brief overview of the HP TechWriter Editor commands, and a detailed description of the syntax and semantics of each command presented in alphabetical order.

Editor Command Summary

Text Modifying Commands

Copy	Inserts text from the copy buffer or an external file in front of the current cursor location.
Delete	Removes text from the current cursor location to the location of the cursor when (Select) key is pressed.
Insert	Inserts text in front of the current cursor location.
Replace	Replaces zero or more of the specified target string with the specified replacement string.
eXchange	Replaces the text at the cursor with text typed from the keyboard, on a character-by-character basis.
Zap	Deletes all text between the anchor and the current cursor location.

Text Formatting Commands

Adjust	Adjusts the column in which one or more lines starts.
Margin	Formats one or more paragraphs according to global or local margins.

Cursor Positioning Commands

Equals	Positions the cursor at the anchor.
Find	Positions the cursor after the specified target string (if search was forward), or on first character of specified target string (if search was backward).
Jump	Positions the cursor at the beginning of the file, the end of the file, or the specified marker.
Page	Positions the cursor ±23 lines from the current location, depending on current direction.

Cursor Keys

Tab	Moves the cursor to next tab position (fixed tabs) in the current direction.
Return	Moves the cursor to the leftmost character of the next line in the current direction.
Space Bar	Moves the cursor one character in the current direction.
Arrow Keys	Moves the cursor in the direction specified by the key.

Document Commands

^^C	Specifies a table of contents title or entry.
^^D	Specifies a drawing to be included.
^^J	Jumps (skips) a specified number of lines in the printed output.
^^L	Turns the output printing on or off.
^^M	Specifies local margins.
^^P	Outputs a conditional or unconditional printer formfeed.
^^S	Specifies line spacing for the printed output.
^^T	Specifies the contents of a page tag (footer).

Miscellaneous Commands

Set	Modifies the environment or sets markers in the text.
Verify	Redraws the screen display.
List	Prints a listing of the file, with or without drawings, on the printer.
Quit	Leaves the Editor in an orderly manner. Provides various ways for saving the text currently in memory.
Stop	When listing, the Stop key terminates the listing, and returns control to the main Editor menu. Otherwise, Stop terminates the Editor program.

Command Syntax and Semantics

The Editor commands are presented in alphabetical order and described in a variety of formats to make them more useful to you. Each command's explanation includes: the command's name, a brief functional description, a diagram showing its legal syntax, the command's prompt (if any) and text which discusses using the command. Each of the command's options are also covered.

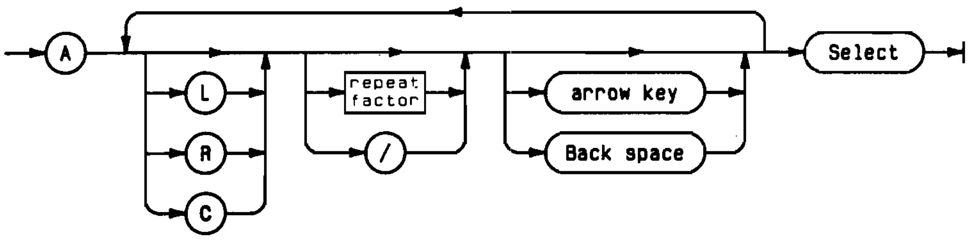
All of the document commands described in this section have headings which imply the command character is the caret (^), although they will work no matter what command character is defined.

Alphabetical Listing of Editor Commands

Adjust
^^C
Copy
^^D
Delete
Equals (=)
Find
Help
Insert
^^J
Jump
^^L
List
^^M
Margin
^^P
Page
Quit
Replace
^^S
Set
^^T
Verify
eXchange
Zap

Adjust

Adjust horizontally shifts the starting column of one or more lines of text.



Item	Variable Parameter Type	Range Restrictions
repeat factor	integer numeric constant	1 thru 9999

Semantics

The Adjust prompt:

```
>Adjust: Left Right Center <arrow Keys> [<sel> to leave]
```

The Adjust command provides a means of adjusting line positions. Adjust uses the line position of the cursor when the command is entered as a starting point. A line-oriented command, Adjust lets you shift an entire line of text to the left or right using the ▶, ◀, or Back space. Repeat factors can be used with these keys to shift the text. For example, pressing 7 ▶ results in the line of text shifting 7 spaces to the right.

Pressing A (for Adjust) and then L, R or C moves the line to the left margin, right margin or centers the line between the two margins. The margins used by these options are the Right and Left margins currently set in the environment (see Set command).

Typing a repeat factor and ▲ or ▼ causes that number of lines to be adjusted in the same way as the accumulated adjustments at that point. The slash (/) functions as an infinite repeat factor and can be used with ▲ and ▼. It causes adjustments to be made from the current line to either the beginning or the end of the text file, respectively. For example, pressing C / ▼ causes all the text between the current cursor position and the end of the file to be Centered according to the current margins.

Note

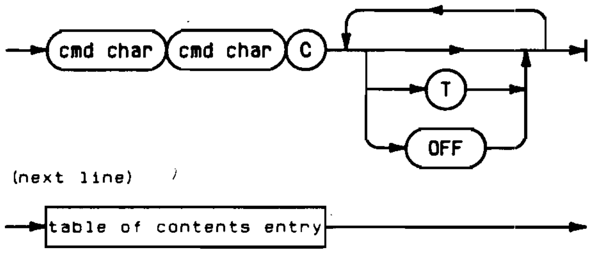
Be careful when using large repeat factors or the slash (/) character to adjust text. This is because the effects of the Adjust command cannot be aborted. Whatever errant adjustments are made will have to be corrected by using the Adjust command again.

Adjust also sets the anchor used by the Zap command. Pressing (=) (the Equals command) moves the cursor to the anchor position.

You must leave the Adjust command by pressing (Select). Aborting with no effect on the text by pressing (Shift)-(Select) is not possible.

^^C

The `^^C` document command specifies a table of contents title or entry. The table of contents can be generated by the Lister program.



Item	Variable Parameter Type	Range Restrictions
table of contents entry	string of ASCII characters	single line; ≤256 characters

Example Document Commands

If the environment's command character is the caret (`^`), and you wish to specify the title of the table of contents, use:

```
^^C T OFF
My Summer Vacation: Volume XVII
```

If `OFF` is not specified, the line will also be included in the listing.

Semantics

If no options are used, the text definition line (the line immediately following the double-command-character line) is merely noted, along with the current page number, and put into the table of contents. The entire contents of the text definition line is printed in the listing of the file. Both titles and entries are only one line long.

If the `T` option is used, the text definition line will appear, above the phrase `Table of Contents`, which appears at the top of each page of the table of contents. If the `T` option is **not** used, the text definition line will appear as an entry in the table of contents, and will be associated with a page number. If the text definition line is blank, a blank line will be inserted in the table of contents.

Whether the `T` option is used or not, the entry will also appear in the text of the document unless the `OFF` option is used.

You can specify the two column labels, "Topic" and "Page", as well as the header, "Table of Contents", by altering the contents of the configuration file (see the section *Customizing the Default Environment* in the *Fully Utilizing HP TechWriter* chapter). These column labels are printed at the right and left margins of the first file in the Lister program file pool. If this file is not a .TEXT file, the margins set by the configuration file are used.

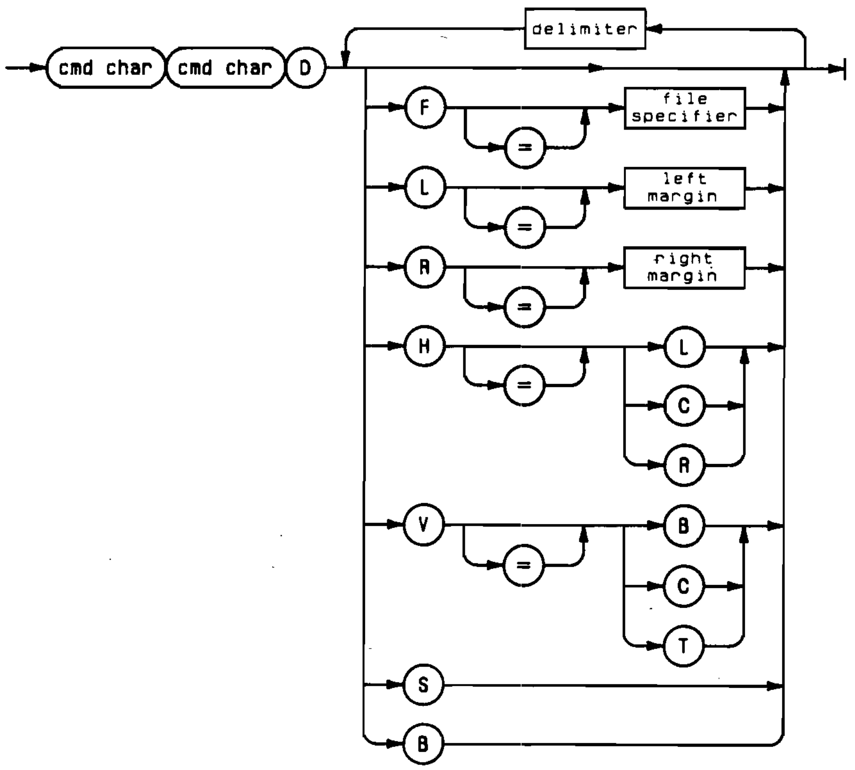
The Lister uses the first tag it encounters while processing the files in the file pool for the table of contents tag. If it does not encounter a tag command, the default tag from the configuration file is used. If you want special margins or a special tag for your table of contents, you may want to make a file with those margins containing nothing but the desired tag to use as the first file in your file pool.

<Page> tokens in tables of contents tags are printed as lower case Roman numerals.

The table of contents will be printed by the Lister program at the end of the document listing, if the tAble of contents parameter is set true.

^^D

The Drawing document command causes a graphics picture to be placed in the text at a specific location.



Item	Variable Parameter Type	Range Restrictions
file specifier	literal; a file name preceded optionally by a volume specifier	no. of characters ≤ 72
left margin	integer numeric constant	0 thru the right margin.
right margin	integer numeric constant	0 thru the number of columns on the screen minus one. The Models 216, 220, and 236 have eighty columns, the Model 226 has 50 columns.
delimiter	ASCII character	space or comma

Example Document Commands

```
^^D, f=#4:Hubcap,L=10,R=69
^^d f=LIST:MyPicture
^^D F=/MainDirectory/Subdirectory/Subsubdirectory/MyPicture,B
^^D f=Drawing 120 r51 s
^^D
```

Semantics

Drawing File Specification

The file specifier consists of a file name and an optional volume specifier. The volume specifier may be a logical unit number (as in the first example above), a volume name (as in the second example above), or an SRM pathname (as in the third example above). If the volume specifier is omitted (as in the fourth example), the file is sought for on the default volume. If the file name is omitted, the specified file cannot be found, or the specified file has not been processed by the *Picture Processor* program, then an empty box is drawn around the limits of the picture.

Note

For a figure to be drawn on the screen, the `Draw figures` parameter in the global environment must be `True`. For a figure to be printed, the `Draw figures` parameter in the List environment must be `True`.

The Drawing Limits

Picture limits are defined as follows. The left and right margins are specified in character cell units. The left edge of the picture is the left edge of the specified left margin character position and the right edge of the picture is the right edge of the specified right margin character position. Left and right margins can be specified in the Drawing command itself, and are considered valid if:

- Both the left and the right margins have values less than the number of columns on the screen. For example, the largest valid value for the Model 236 computer is 79 because its screen is 80 columns wide. 79 is the maximum number because the columns are numbered 0 through 79.
- The left margin to be used is less than or equal to the right margin to be used. The margins "to be used" are margins in the `^^D` command if they are specified, or global margins from the environment, otherwise.

If the left/right margin values in the `^^D` command are invalid, they are ignored, and global margins are used.

Top and bottom limits are determined solely from the number of lines which exist in the text file between the Drawing command and the next command line. The picture-terminating command line may be an empty command: e.g., `^^`. The maximum height for a drawing is 21 lines (a full screen), and the minimum height is one line.

Justifying a Drawing Within Its Limits

If a picture is drawn which does not fill the entire drawing area, there is some room left over either on the right, left, top, or bottom. You can specify the justification of the picture within the drawing limits. The options to the Draw command which do this are:

- To horizontally justify the picture, specify `H=L`, `H=C`, or `H=R` to left-, center-, or right-justify the picture within the drawing limits, respectively.
- To vertically justify the picture, specify `V=B`, `V=C`, or `V=T` to bottom-, center-, or top-justify the picture within the drawing limits, respectively.

Both horizontal and vertical picture justification are centered if you do not specify otherwise.

The picture-justification capability is especially useful when sending drawings to printers whose dot size, resolution, and width:height ratio are different from those of the screen. The justification parameters tell the Editor how to position the drawing with respect to the text columns and lines around it.

Anisotropic Drawings

By default, the figures are drawn isotropically; that is, one unit in the horizontal (X) direction is the same size as one unit in the vertical (Y) direction. Typically, this is what is desired, and therefore is the default. If you want anisotropic scaling, where one unit in the horizontal (X) direction is *not* necessarily the same size as one unit in the vertical (Y) direction, specify an *S* in the Draw figure command, e.g., `^^D,F=LIST:mech,l=20,r=50,S`. The *S* means *stretch* the drawing to completely fill the available area, as defined by the starting and ending lines, and the left/right margins (local or global).

Drawing a Frame Around a Drawing

If the `B` option is used, a box is drawn around the periphery of the picture limits.

Drawing Size

A picture must be able to fit entirely on the screen at one time to be seen, both in the Editor, and in a hardcopy listing.

Aborting a Drawing

A figure in the process of being drawn can be aborted by pressing (Shift)-(Select). The drawing process is immediately stopped *if the* (Shift)-(Select) *was the first thing in the type-ahead buffer.* If it is not the first thing, the drawing process will not stop. Pressing (CTRL)-(Clear line) will clear the type-ahead buffer. Then, (Shift)-(Select) will be the first thing in the type-ahead buffer, and thus will abort the drawing. If you find out you do want the picture drawn, you can scroll it off the screen and then back on, or just press (V) ("Verify") to redraw the screen.

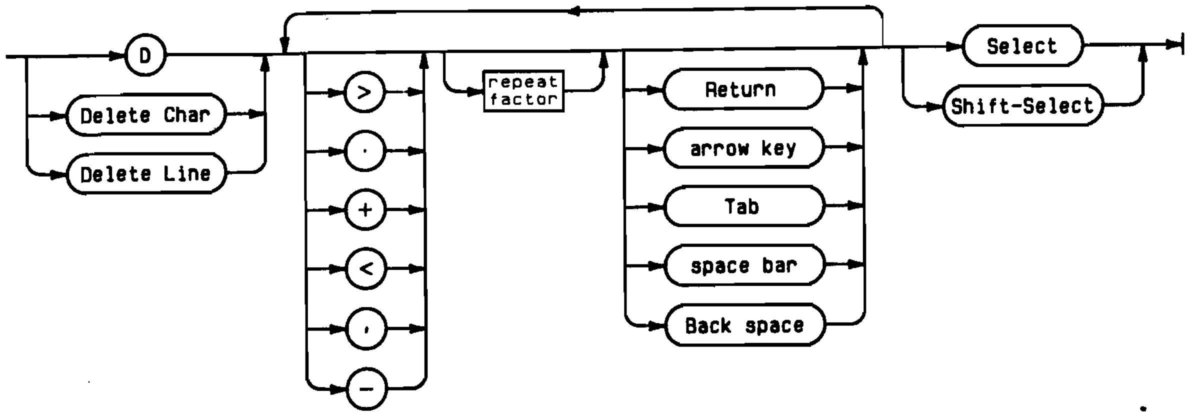
Drawing Error Messages

There are several error conditions which can exist in a file with drawing commands. If an error condition exists, *and* if `Draw figures` in the environment is `True`, the Editor displays an error message on the same line as the Drawing command. It is displayed in inverse video if your screen supports this. *The messages are only telltales from the Editor; they are not put into your file.* Several error conditions can exist:

- If there are no lines between the Drawing command line and the terminating command line, the error message is `NO ROOM FOR FIGURE.`
- If there are more than 21 lines between the Drawing command line and the terminating command line, the message is `FIGURE WON'T FIT ON CRT.`
- When you are scrolling through your file (the text scrolling *up*), and a drawing command scrolls onto the screen, but the terminating command (twenty-three or less lines later) has not yet done so, another message appears. This particular one is not an *error* message, per se, since the figure can fit on the screen if you just scroll some more, but is more a *status* message: `FIGURE WON'T FIT ON CRT.`
- If a drawing is not terminated (that is, the end of the file is found before the end of the figure), this message appears: `FIGURE MUST BE TERMINATED.`

Delete

Delete removes text from the current file.



Item	Variable Parameter Type	Range Restrictions
repeat factor	integer numeric constant	1 thru 9999 (1 thru 4095 for (Tab))

Semantics

The Delete prompt:

```
>Delete: < > <arrow keys>      [<sel> deletes, <sh-sel> aborts]
```

The Delete command enables you to remove text and fills the copy buffer with the deleted text. Delete uses as a starting point the cursor position when the command is entered. Subsequent cursor movement by any cursor control key causes text to be removed between this point and the new cursor position. Text can be recovered by moving the cursor back toward the starting point.

Direction applies in the Delete command and is shown by (>) (forward) or (<) (backward) in the Delete prompt. If the direction is forward, movement caused by the space bar, (Tab), or (Return) occurs from the cursor toward the end of the file; if it is backwards, movement is from the cursor toward the beginning of the file. Motion caused by the arrows keys and backspace is always intuitive regardless of the direction indicator. Direction can be changed while in the Delete command by pressing >, ., or + (for forward) or <, ,, or - (for backward).

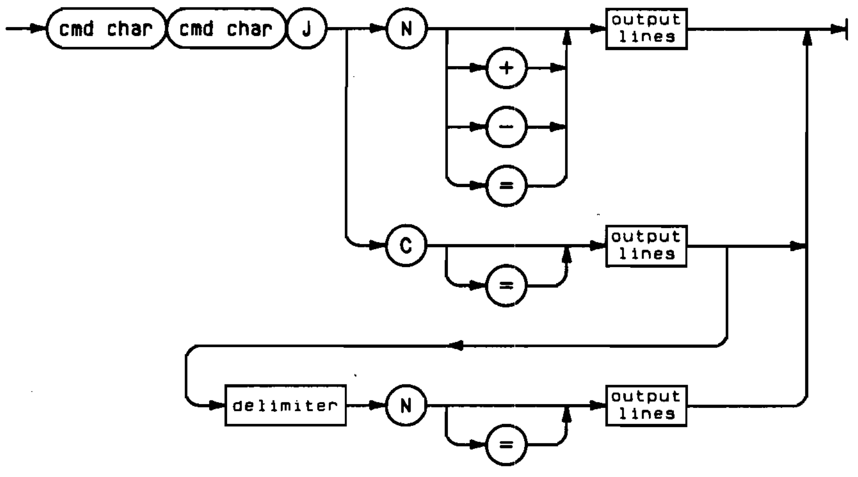
Repeat factors are available within the Delete command. For example, pressing (D) (for Delete) and then 9 (Return) removes 9 lines of text in the current direction starting at the cursor position.

To exit the Delete command press (Select) or (Shift)-(Select). (Select) confirms the deletion, returns the Editor prompt and displays the cursor at its position when (Select) was pressed. (Shift)-(Select) aborts all changes made since Delete was entered, returns the Editor prompt and displays the cursor at its position when Delete was entered.

The copy buffer (see the Copy command) is filled by the deleted text, regardless of whether the command is exited with (Select) or (Shift)-(Select).

^^J

The ^^J document command jumps, or skips, output lines.



Item	Variable Parameter Type	Range Restrictions
output lines	integer numeric constant	0 thru the number of lines per page
delimiter	ASCII character	space or comma

Example Document Commands

```
^^J  N=10
^^J  c15
^^j  C=12, N=4
^^j  N+8
^^j  N-6
```

Semantics

The ^^J command allows you to skip any number of lines on the output up to the number of lines on a page. The number of lines on the page is equal to `Number of lines per page` minus `Top lines skipped`, and, if `Include Tag` is true, minus three more. If you specify zero lines, the command does nothing.

For the following discussion, C is the number of lines associated with the condition, and N is the number of lines specified to skip. So, for the third example line above, C equals twelve and N equals four.

If N is present, and C is not, the computer looks at the sign specified for the number of lines to skip. If it is +, the computer will print the specified number of blank lines. If it is =, the computer will print the specified number of blank lines *in one block*, go to a new page first if necessary. If the sign is -, the computer will print blank lines until there are N lines left at the bottom of the page. If there are not N lines left on this page, a new page will be generated before the lines are skipped.

If C is present, and N is not, the specified number of lines will be skipped, or the computer will go to the top of a new page, whichever comes first.

If both C and N are present, the computer checks the value of C. If the number of lines specified for C remain on the page, a block of N contiguous lines is skipped. The sign on N is assumed to be =.

If neither C nor N is specified, nothing is done.

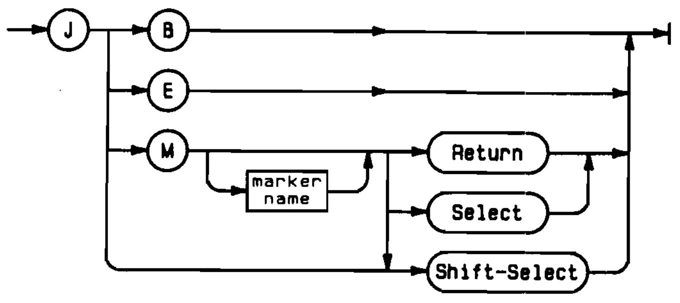
These conditions and operations are summed up by the following logic structure of the Jump command:

```
Is C present?
    If so, is required lines greater than lines left?
        If so, then go to the top of the next page.
        If not, is N value present?
            If so, skip N lines, ignoring sign.
            If not, skip C lines.
    If not, is N present?
        If so, what is the sign?
            +) Skip N lines.
            =) Is N greater than lines left?
                If so, new page, then skip N lines.
                If not, skip N lines.
            -) Is N greater than lines left?
                If so, new page, then skip (total lines) - N lines.
                If not, skip (total lines)-(current line) - N lines.
        If not, do nothing.
```

The ^^J Jump command *cannot* be used to jump over lines which will be filled with a drawing drawn by the ^^D Drawing command, because the Drawing command is terminated by any double-command-character line, including a ^^J command.

Jump

The Jump command repositions the cursor.



Item	Variable Parameter Type	Range Restrictions
marker name	literal	1 to 8 ASCII characters excluding: CHR(0) thru CHR(31), blank, comma, and CHR(127)

Semantics

The Jump prompt:

```
>JUMP: Begin End Marker  [<sh-sel> to leave]
```

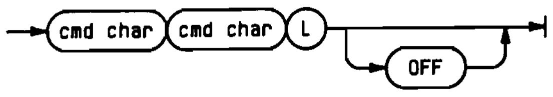
The Jump command moves the cursor to the beginning or end of a text file or to a previously defined marker. The command has no other effects; it merely repositions the cursor. To Jump to the beginning of your file, press J B. To Jump to the end of your file, press J E.

You can also Jump to a marker by pressing J M and typing the name of any previously set marker in the file followed by Return. Marker names are defined with the Set command.

Marker names are not case sensitive. The Editor converts all marker names to upper case letters so they can be typed using any desired combination of upper case and lower case letters. See the Set command for more information on markers.

^^L

The `^^L` document command disables and enables listing.



Example Document Commands

```
^^L
^^L off
```

Semantics

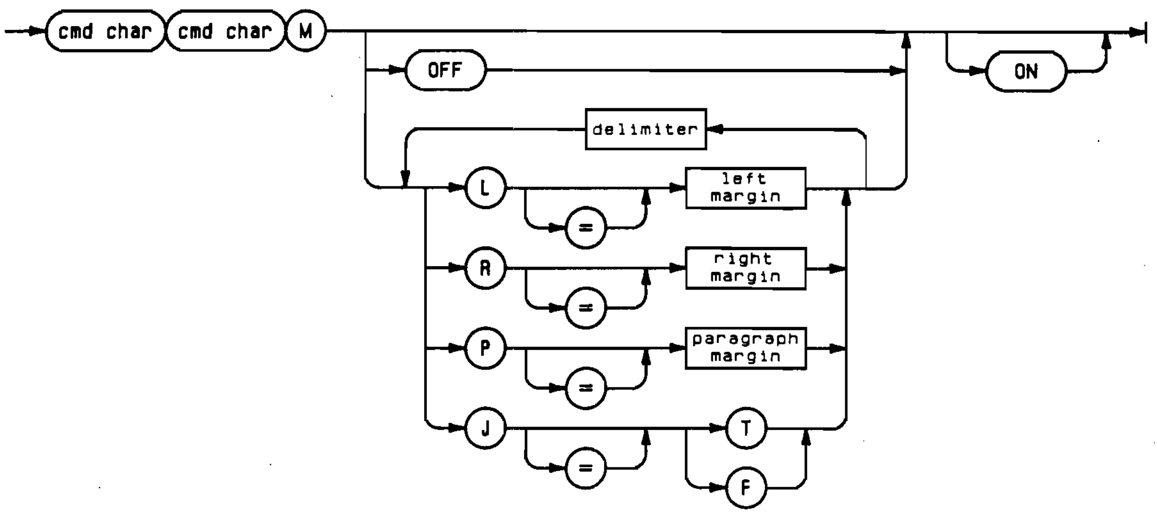
The `^^L off` command turns off listing for all the following text until a `^^L` or `^^L on` command is found, which turns listing back on. If listing is turned off, and is not turned back on, none of the remainder of the file is printed.

All document commands in a section of a document which is not being printed are ignored. A `^^L` command in the file cannot be overridden from the List menu.

A list command turning on or off the listing when the listing is already in that state is ignored. That is, an `^^L off` immediately following another `^^L off` is ignored; listing is *already* off.

^^M

The ^^M document command specifies the margin specifications for the following paragraph. The margin specifications may be different than those set in the environment. If ON is specified, the local parameters are in effect until explicitly reset by another ^^M document command.



Item	Variable Parameter Type	Range Restrictions
left margin	integer constant	0 thru right margin minus one
right margin	integer constant	1 thru 255
paragraph margin	integer constant	0 thru right margin minus one
delimiter	ASCII character	space or comma

Example Document Commands

```
^^M
^^M,l=10,r=50
^^M p0 l0 r65 jt on
^^m off
^^m off on
^^m L=23,J=F ON
```

Semantics

If no options are specified in a `^^M` document command, as in the first example above, the command instructs the editor to use the global values. This form of the `^^M` document command is particularly useful for terminating the effect of a previous `^^M` document command with an "ON" option (described below).

Left, right, and paragraph margins as well as the Justification switch may be specified in the command. If these are valid, they will be used. If not, the global values from the environment will be used. Invalid values could be:

- A left or a paragraph margin greater than the right margin,
- A value for Justify (`J`) which is neither `T` nor `F` (case ignored), or
- A token which is unrecognized, e.g., `M of`.

Local values are used if specified, but all do not have to be specified. Therefore, you could specify a local left margin but not specify a local right or paragraph margin. In this case, margining would use the global paragraph and right margin values, and the local left margin (if valid).

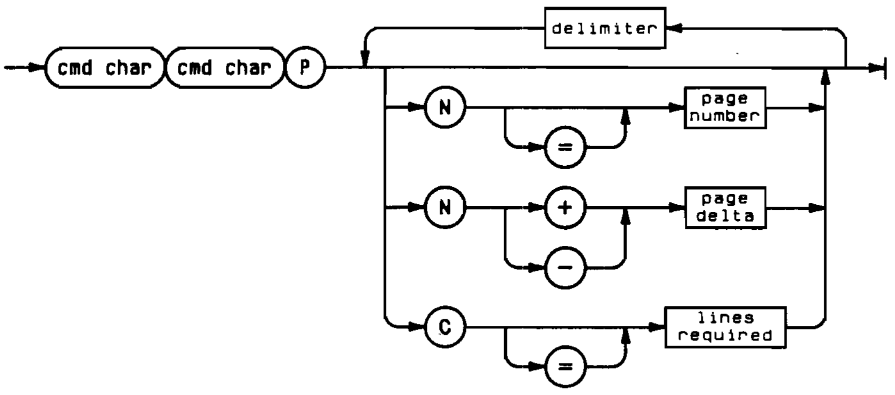
The keyword `OFF` may be used to temporarily suspend the effects of margining for the "paragraph" following the document command. That is, the text format of a paragraph (see the `Margin` command) that begins with a "^^M OFF" document command will not be altered by a `Margin` command, or an `Insert` command done while in document mode.

The keyword `ON` may be used to invoke "locking margins," that is, margins which are in effect until explicitly reset by a subsequent `^^M` document command. Similarly, margining of paragraphs can be disabled by using a `^^m off on` document command.

The equals signs separating the parameters from their values are optional.

^^P

The `^^P` document command gives you a conditional or unconditional page break, or sets the current page number to the specified value.



Item	Variable Parameter Type	Range Restrictions
page number	integer numeric constant	0 thru 9999
page delta	integer numeric constant	0 thru 9999
lines required	integer numeric constant	0 thru the actual number of lines per page
delimiter	ASCII character	space or comma

Example Document Commands

```
^^P
^^P N=14
^^P n=8,c=0
^^P c23
^^P N+5
```

Semantics

The document command `^^P` with no optional parameters simply causes an unconditional page break. The listing immediately goes to the top of a new page, and the page number is incremented.

Using the `C` option specifies a conditional new page: a new page will be started if there are less than the specified number of lines left on the current page. For example, `^^P c=12` tells the Editor to go to the top of a new page if there are less than twelve lines left on the current one. If the number of lines specified is greater than the number of lines on a page, the conditional page command option will be ignored.

The N option allows you to alter the page number. Absolute specification is achieved through specifying an unsigned page number, e.g., `^^P n=45` means, "go to the top of a new page, and call it page 45."

Relative page number specification is achieved through specifying a signed page number. For example, `^^P n+10` issues a pagefeed and makes the page number ten greater than it was. Similarly, `^^P N-12` issues a pagefeed and decreases the page number by twelve.

If you wish to modify the page number without issuing a pagefeed, use a conditional page command where the condition is always false: `^^P n=34,c=0`. This sets the page number to 34 and would issue a pagefeed if there were less than zero lines left (there are *never* less than zero lines left on a page).

Note

If a Find has been done since the last Replace, the target string used by the "same" option is now the target specified in the Find command.

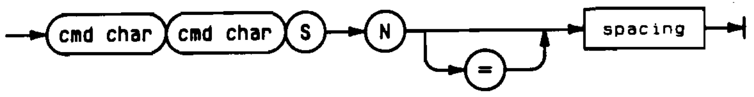
The **Ignore case** option in the environment applies to the Replace command. Searches are sensitive to the case of the characters (upper/lower) unless `Ignore case` is set to `True` in the Environment. When a match is found, the target string is replaced with the replacement string. The case of the replacement string is not affected by the `Ignore case` option.

The anchor (used by the Zap and Equals command) is set at the location of the most recent string found, whether it is replaced or not.

The Replace command can be aborted before all specifications are complete by pressing (Shift)-(Select). If (Shift)-(Select) is used, the target and replacement patterns used by the "same" option remain unchanged. (Select) cannot be used with the Replace command. The command is executed immediately when the final delimiter (or (S)) is typed.

^^S

The Spacing document command specifies the spacing of lines in the output listing.



Item	Variable Parameter Type	Range Restrictions
spacing	integer numeric constant	1 thru the actual number of lines per page minus 1

Example Document Commands

```
^^S N=0
^^s N=3
^^s
```

Semantics

The Spacing document command specifies the number of carriage-return/linefeeds between contiguous lines output to a printer. If you set *n* equal to 1, you get single-spacing; if you set *n* equal to 2, you get double-spacing; etc. An *n* of zero causes overprinting.

When *n* is not included in the command (as in the last example above), a value of 1 is used.

The Right margin must be greater than the Left margin setting or an error message is generated when you attempt to define that parameter:

```
>ERROR: Improper margins <space> continues.
```

Para margin

The **Paragraph margin** can be set to any positive integer less than `Right margin`. This setting defines the indentation of the first line of each paragraph.

Indentation is relative to the first column, not the `Left margin` parameter. Thus, if `Left margin` is 10 and `Para margin` is 8, the first line of each paragraph would start two columns to the *left* of the rest of the lines.

Indentation occurs while inserting text when `Auto indent` is set `False` and `Filling` is `True` or when Margin is used. Note that a paragraph as defined by the Editor is any text surrounded by blank lines or by lines beginning with the Command character (discussed below). The beginning or end of a file will also delimit paragraphs.

The Para margin must be less than the Right margin setting or an error message is generated when you attempt to define that parameter:

```
>ERROR: Improper margins <space> continues.
```

Command ch

The **Command character** can be any non-control, non-blank character. If this character is the first non-blank character in a line, the Margin command treats the line as if it were blank. The line is not margined and it is considered to be the beginning or the end of a paragraph. If the first two characters on a line are command characters, the line is assumed to be a document command and is interpreted accordingly.

During a listing, a single command character at the beginning of a line is replaced by a blank. Also, unknown commands have their first character replaced by a blank during listing. The default command character is the caret (^) character.

The command character cannot be the same as the underline character or the escape character.

Underline ch

When making a listing, either from the Editor or Lister (see the *Lister Reference* chapter), it is possible to underline text in the hard copy. This is done by placing occurrences of the underline character in your text. The first occurrence of the underline character initiates underlining, and the second occurrence turns it back off. The third occurrence turns it back on, and the fourth turns it off again, and so forth. A file, therefore, should have an even number of underline characters in it. If it does not, your file listing will probably not look like what you desire. If there is an odd number of underline characters in a file, underlining will be turned off when the end of the file is reached.

Occurrences of the underline character are ignored when counting characters for margining. Thus, when margining, lines containing the underline character may be longer than they look like they're supposed to be. These will be "swallowed" during hard-copy listing, so the margining behavior concerning underline characters is correct for listing, but not for the soft copy on the screen.

The underline character cannot be the same as the command character or the escape character, but it may be blank, indicating that there is none.

Escape ch

This character enables you to send any byte value you wish to the printer you're using. After this escape character is defined, you can use it in your text, followed by three (base ten) digits indicating the byte value you wish to send. The number must be between 000 and 255, inclusive, or it is ignored. The functionality is much the same as the use of the **ANY CHAR** function accessed through the softkeys.

These groups of four characters are ignored during margining, the same as the underline character. Thus, a line containing one or more of these escape sequences will extend farther to the right than it "should."

If you use escape sequences to access special printer capabilities, the printer is **not** brought back to its default state after the listing is completed.

The escape character cannot be the same as the command character or the underline character, but it may be blank, indicating that there is none.

Draw figures

Draw figures is a true or false value. When it is set to `True`, any figures specified by `^^D` document commands which can fit on the screen will be drawn.

If a figure is being drawn, and you do not want to wait for it to complete, press (Shift)-(Select). This will abort the drawing, if there is nothing in the type-ahead buffer in front of the (Shift)-(Select). If there is, press (CTRL)-(Clear line), to clear the type-ahead buffer, and then (Shift)-(Select).

Token def

Token def is a true or false value used by the Find and Replace commands. When `Token def` is set `True`, the Find and Replace commands perform token searches by default. Conversely, when Token def is set `False`, the Find and Replace commands perform literal searches by default. (See the Find or Replace command for more information.)

Ignore case

Ignore case is a true or false value used by the Find and Replace command. When `Ignore case` is set `False`, then "`string`", "`STRING`", and "`String`" are all considered to be different. If `Ignore case` is set to `True`, they are treated as equal. (See the Find or Replace command for more information.)

Zap markers

The **Zap markers** command removes all markers from the file.

File Type

In the lower right corner of the Set Environment display, the Editor indicates what kind of file you are editing, whether it be `TEXT`, `ASC`, `DATA`, or `new`. A `new` type is used for text data which has not yet been stored as a file.

The environment display is exited and the Editor's main prompt returned by pressing (Return), (Select) or the space bar. The current environment settings are automatically saved with your file when the text is written *as a* `.TEXT` *file* to a disc or other mass storage medium. If you write your file as a `Data`-type file (by suffixing the file name with a period) or as an ASCII file (by suffixing the file name with `.ASC`), the environment will *not* be saved.

Text Entry Modes

There is only a single environment associated with a text file at any time. There are two standard text entry modes provided by the environment:

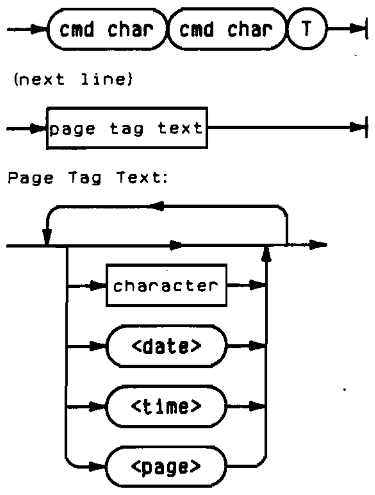
Program mode	This sets the environment for writing programs or tables and may be selected by pressing (S) (P). This makes `Auto indent True` and `Filling False`.
Document mode	This sets the environment for writing non-program and non-tabular text, and may be selected by pressing (S) (D). This makes `Auto indent False` and `Filling True`.

When either predefined environment is set, the current environment is displayed and any of its parameters can be changed. If you want to change other parameters, use (S) (E) to get into the existing environment.

Changes made to the environment cannot be aborted but the parameters may be changed as many times as desired.

^^T

The ^^T command causes a specific page tag, or footer, to be printed at the bottom of each page of a listing.



Item	Variable Parameter Type	Range Restrictions
page tag text	string literal; default = "Page <page>"	may contain any non-control ASCII characters
character	ASCII character	any non-control ASCII character

Example Document Commands

```
^^T
This is my page footer.  Date: <date> Time: <time>

^^T
--<page>--

^^t
```

Semantics

The `^^T` command takes the next line of text and causes it to be printed at the bottom of every page of a listing.

There are three special tokens whose values will be placed in the footer at the time of listing. These tokens must be either all upper, or all lower, case characters, and must have no embedded blanks:

`<date>`	The six characters "`<date>`" will be replaced by the current date at the time the listing was initiated. The date will be in the form mM/dD/YY, dD/mM/YY, YY/mM/dD, etc., whatever is specified in your configuration file. The lowercase character in "mM" and "dD" indicate those spaces will be used only if necessary; leading zeroes will not be printed, except on the year. In all cases, there will be six to eight characters.
`<time>`	The six characters "`<time>`" will be replaced by the current time at the moment the List command was invoked. The time will be in the form HH:MM, and hours range from 0 through 23 (24-hour time).
`<page>`	The six characters "`<page>`" will be replaced by one or more characters, depending on the current value of the page number. The page number will have no leading or trailing blanks.

The date and time placed in the footer will be the system date and time at the point when the List command was entered; thus, the time printed in each page's tag will not differ from that of the previous page.

Specifying a `^^T` command followed by a blank line, causes the page tag to be set to a blank line. If you do not want a page tag to be printed at all, then set the `Include Tag` to `False` in the List environment.

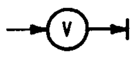
The `^^T` command is a two-line command, regardless of the content of the second line. For example, consider the following text in a file:

```
^^T
^^L off
```

The `^^L off` command will go into the footer and *will not* turn off the listing.

Verify

Verify refreshes the screen display from memory.

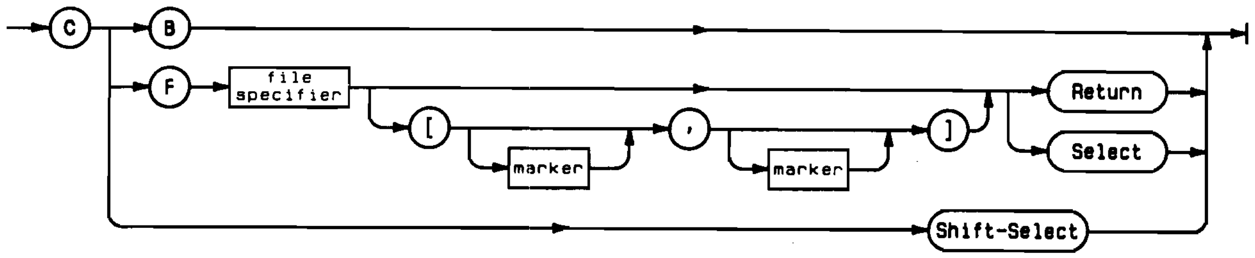


Semantics

The Verify command has no options; it is executed immediately by pressing $\boxed{\text{V}}$. Verify causes the Editor to refresh or update the screen with the current text (if any) from memory, and the current figures (if any) from mass storage, and to display the Editor prompt.

Copy

Copy inserts text from a specified file or from the copy buffer.



Item	Variable Parameter Type	Range Restrictions
file specifier	literal	any valid file specifier
marker	literal	1 to 8 ASCII characters excluding CHR(0) thru CHR(32), comma, and chr(127)

Semantics

The Copy prompt:

```
>Copy: Buffer File  [<sh-sel> to leave]
```

The Copy command provides a way of moving or duplicating text in a file and copying text from another file. These are the Buffer and File options. Pressing $\boxed{C}$ (for Copy) and $\boxed{B}$ (for Buffer) results in the contents of the copy buffer being written at the cursor position when the command was entered. The screen displays the new text and the Editor prompt.

The copy buffer is filled with the text involved in the most recent Delete, Insert or Zap command, and its contents are cleared by the Copy File and Margin commands. A Copy Buffer command after the copy buffer has been cleared generates the message:

```
>ERROR: Invalid copy.  <space> continues.
```

Doing a Copy Buffer does *not* alter the buffer's contents, nor do any cursor movement operations. Thus, you can make multiple copies of the same text in different locations by repeatedly positioning the cursor and pressing C B.

To Copy from a file, press C F. The screen displays:

```
>Copy: File[marker,marker] ?
```

The Editor is requesting a file name, and two (or less) marker names. The volume name may be omitted if the file in question is on the default volume. Specification of the previously set markers (see Set command) is optional but, if given, the marker names must be enclosed in square brackets and separated by a comma. See the following examples; portions of a file called `MyFile` on the default volume are being copied in:

`MyFile`	(whole file)
`MyFile[,]`	(whole file)
`MyFile[]`	(error)
`MyFile[A]`	(error)
`MyFile[,B]`	(from beginning of file through marker B)
`MyFile[A,]`	(from marker A to end of file)
`MyFile[A,B]`	(from marker A through marker B)

Remember, only `.TEXT` type files have markers.

The copy inserts the file's contents at the cursor's position when the Copy command was entered, and when the copy is complete the cursor is placed at the first character of the copied text. You can exit the command before all specifications are complete by pressing Clear line, Return.

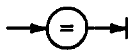
After typing the appropriate information and pressing Return, the Editor displays:

```
>Copy...
```

This shows that the specified text is being copied into your current text. When the operation is complete, the Editor prompt reappears and the screen displays all or part (depending how much was copied in) of the text that was copied.

Equals (=)

$\boxed{=}$ positions the cursor at the anchor's location.



Semantics

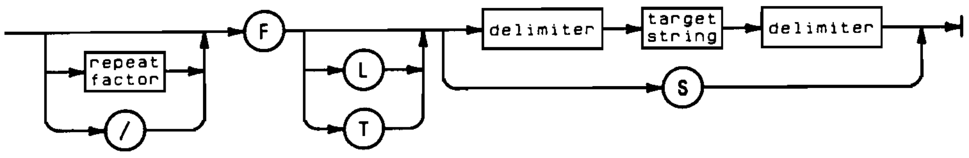
The equals sign (=) is a cursor-positioning command. It moves the cursor to the beginning of the most recent item Found, Adjusted, Inserted, or Replaced. Pressing $\boxed{=}$ causes the cursor to jump to the location of this "anchor" and the Editor's prompt is displayed. This is the anchor used by the Zap command.

These are the rules whereby the anchor is set:

Find:	The anchor is set at the first character of the most recently-found string. If the string being sought was not found, the anchor is not moved from its previous position in the file.
Adjust:	The anchor is set at the position of the cursor at the time the Adjust command is invoked. This is always true, whether the line is ultimately moved or not.
Insert:	For Insert, the anchor is set at the first character of the string that is inserted. If no text is inserted, the anchor is set to the cursor position at the time Insert is invoked.
Replace:	The anchor is set at the first character of the most recently found string. If the string being sought was not found, the anchor is not moved from its previous position in the file.

Find

Find moves the cursor to an occurrence of a specified string.



Item	Variable Parameter Type	Range Restrictions
repeat factor	integer numeric constant	1 thru 9999
delimiter	literal	any valid delimiter; must be used in matched pairs
target string	literal	1 thru 128 characters

Semantics

The Find prompt:

```
>Find[1]: L <target>=>
```

or

```
>Find[1]: T <target>=>
```

The prompt displayed depends on whether the "Token" definition in the Editor's environment is set to `True` or `False`. If it is set to `True`, the first prompt is displayed; if it is `False`, the second is shown. These are explained below.

In its simplest form, the Find command is executed by pressing F and specifying a string surrounded by delimiters. Upon typing the final delimiter, the cursor is positioned at the end of the first occurrence of the specified string in the current direction.

The Find command moves the cursor and sets the anchor (used by Zap) at the location of the target string. In this context, a "string" is a contiguous series of non-control ASCII characters surrounded by delimiters. A delimiter is a separator that signifies to the Editor the beginning or end of the string and it must be one of the following characters:

```
! " # $ % _ ' ` + - * / ^ = . , : ; < > ( ) [ ] { } | \ ~ @ & ?
```

The cursor position when finding is taking place in the forward direction (>) is immediately to the right of the rightmost character in the string found. The cursor position when finding is taking place in the backward direction (<) is under the leftmost character in the string found.

Don't use a delimiter that appears in your string. Delimiters must be matched pairs; if you use "$" to signify the beginning of a string, you must use "$" to signify the end of the string.

The maximum length of a target string is 128 characters. If you attempt to find a null string, i.e., a delimited string with zero characters between the delimiters, the main Editor prompt line returns, and the cursor will not have moved. If you specify a string which cannot be found, the Editor responds with:

```
>ERROR: Pattern not found. <space> continues.
```

When a pattern is not found, and you press the space bar to continue, the cursor will not have moved from its position before the command was invoked.

If you try using an invalid character for a delimiter, the Editor responds:

```
>ERROR: Invalid delimiter. <space> continues.
```

The Find prompt shows the current direction. When searching for a string occurrence, Find looks for that string between the cursor position when the command was entered and either the end of the file (if direction is forward) or the beginning of the file (if direction is backward).

Repeat factors are available with the Find command but must be typed before the Find command is initiated. If a repeat factor is used, the Editor positions the cursor at the end of that occurrence of the string. For example, typing 8 F /the/ results in the cursor being positioned at the end of the eighth occurrence of the. The slash (/) operates in a similar way but signifies the last occurrence of the specified string in the current direction. If no repeat factor or slash character is specified it defaults to 1 and the Editor attempts to find the first string occurrence. The Find prompt displays this value in brackets.

If a finite repeat factor is used, and at least one occurrence of the target string is found, but not as many as you wanted, the Editor beeps a displays a message similar to the following:

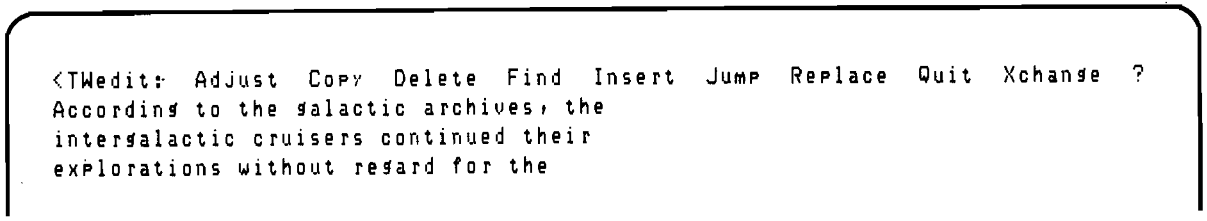
```
>WARNING: Found 35, <space> continues.
```

Press the space bar to continue the editing session.

After pressing F, the prompt on your screen contains either an L or T for "literal" or "token" modes. Literal and token are interdependent; if one option is shown as available, the other is automatically the default. If L is shown in the prompt and you want to use the token search mode, simply type in the target string surrounded by delimiters. The search takes place in the default mode (in this case, token). To do the same search in the literal mode, press L then type in the string as before. The Find command then searches for a literal form of the string.

A **literal string** is a string of characters either isolated or embedded in a word or line. A **token string** is one which is isolated by delimiters, which are any ASCII characters except numbers or alphabetic characters. Blanks are common delimiters in English text because they separate words.

The "**same**" option allows you to utilize the most recent target string, used in either the Find or Replace command, without re-typing it. Suppose you have the following text on your screen with the cursor in the word "explorations" and you type the sequence F L `*galactic*`.



```
<TWedit: Adjust Copy Delete Find Insert Jump Replace Quit Xchange ?
According to the galactic archives, the
intergalactic cruisers continued their
explorations without regard for the
```

After pressing the final delimiter (*), the Editor moves the cursor to the first character of the first literal occurrence (in the backward direction) of the target string `galactic`: the one in the word "intergalactic". Then typing F L S results in the cursor moving to the first character of the next literal occurrence of the same target: the word "galactic" immediately before "archives".

Note

If a Replace has been done since the last Find operation, the target string used by the "same" option is now the target most recently specified in the Replace command.

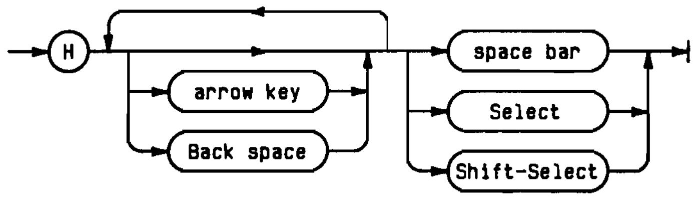
Searches are sensitive to the case of the characters (upper/lower) unless `Ignore case` is set to `True` in the Environment.

Find sets the anchor used in the Zap command and accessed by the Equals command.

Shift-Select can be used to abort the Find command before all specifications are complete. If Shift-Select is used, the target pattern used by the "same" option remains unchanged. Select cannot be used with the Find command. The command is executed immediately when the final delimiter (or S) is typed.

Help

Help fills the screen with text that briefly describes the commands and their operations.



Semantics

The Help prompt:

```
>Help 1:  <arrow Keys>  <sel> or <sp> leaves
```

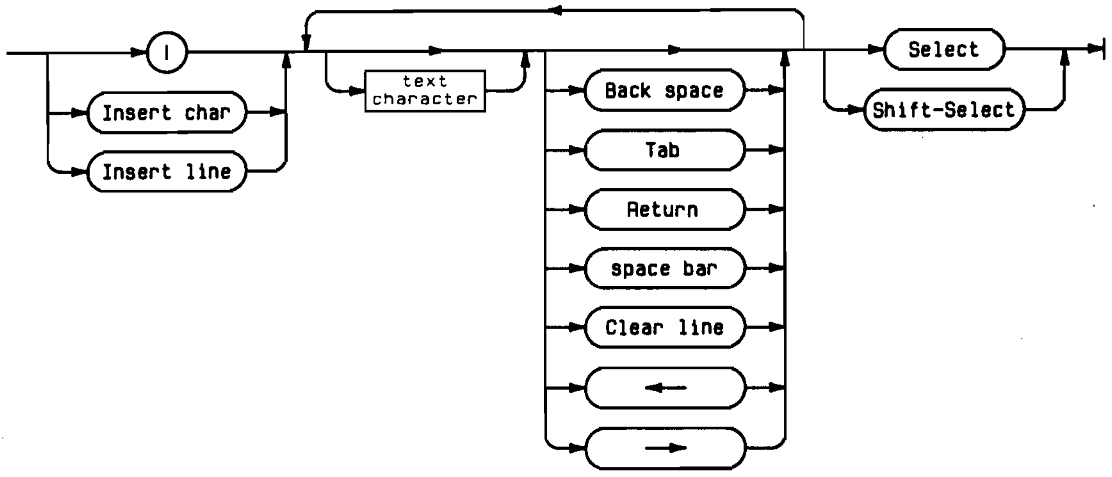
The Help command erases the text display from the screen and replaces it with the text that describes the commands available from the Editor.

Direction does not apply in the Help command. The number following the word "Help" in the prompt indicates which screen of help information is currently being displayed.

Press ▶ to view the next screen of help information, and ◀ to view the previous screen of help information. You can alternate between the help displays as long as you wish. Press Select to erase the help screen currently being displayed and re-display the text being edited, with the cursor positioned where it was when H was pressed.

Insert

Insert opens a window in the current file for the subsequent insertion of text.



Item	Variable Parameter Type	Range Restrictions
text character	literal	any valid ASCII character excluding: chr(0) thru chr(31) and chr(127)

Semantics

The Insert prompt:

```
>Insert: <text> <bs> <clr ln>  [<sel> inserts, <sh-sel> aborts]
```

The Insert command opens a window in the text file directly in front of the cursor position for text creation. When initiated (by pressing I or Insert char or Insert line), the text from the cursor to the right end of the line is shifted to the right edge of the screen. If there is no room (i.e., that line already extends past the right edge of the screen), all lines below the cursor line on the screen are erased, and the remainder of the line of text is placed on the line below.

Insertion always takes place directly in front of the cursor location at the point when Insert was entered. Any sequence of non-control ASCII characters may be inserted and any non-vertical cursor control key may be used. If the cursor is moved backward, text is erased. If the cursor direction is forward, spaces are generated. You can Backspace to delete a character or press Clear line to delete to the end of the most recently inserted line. Clear line is available only after a line of text has been inserted. Backspacing past the point at which Insert was entered is not possible.

The way in which text insertion takes place depends on parameters set in the Editor's environment. These parameters have default values supplied by the Editor but can be changed with the Set command. The ones that concern you here are `Auto indent`, `Filling` and `Justify`. Most of what you need to know about `Filling` and `Auto indent` can be summed up as follows: if you are writing program source text or tables, set Program mode; if writing regular text, set Document mode. Program mode is set by pressing (S), (P), space bar, and defines `Auto indent` to be `True` and `Filling False`. Document mode is set by pressing (S), (D), space bar, and defines `Auto indent` to be `False` and `Filling True`.

`Filling`, when set `True`, performs a *word wrap* function. As inserted text approaches the Right margin (another environment option), the Editor attempts to fit the words on the current line. If a word would cause the line to extend beyond the right margin, as set in the environment, it is automatically moved to the next line (i.e., the system supplies a carriage return and a line feed). When the insertion is completed by pressing (Select), all of the inserted text is margined, if Document mode is set. During this final margining operation, margin parameter values specified with local margin document commands will be honored (only the environment's margin parameters are recognized prior to the insertion command's completion).

Margining with `Justify` set `False` adjusts the text to fit between the left and right margins and also compresses blanks in the text. Margining with `Justify` set `True` adjusts the text to fit between the left and right margins and adds as many blanks as necessary between words to position the end of each line at the right margin. The addition of "justification blanks" begins at the opposite end of every other line, resulting in paragraphs with a more uniform "density" (that is, one side of the page doesn't have a higher incidence of added blanks than the other).

Two blanks can follow the seven characters: `! . ; : ? " )`. All other blanks can be compressed into a single blank character. "Can be," rather than "are," because if a paragraph is margined when `Justify` is `True`, spaces may be added after other characters to make a smooth right margin, but they will be collapsed again if the paragraph is subsequently margined with `Justify False`. However, double spaces following the aforementioned seven characters will *not* be collapsed.

Note that the Editor's definition of a paragraph is *any* text delimited by *any* combination of blank lines, lines having the Command character as the first non-blank character in a line, or the beginning or end of a text file. The Command character is yet another of the environment's options; see the Set command for more details.

As the definition of a paragraph infers, the Editor does not distinguish tables from other kinds of text material. Any insertions within a table will result in the table being margined (i.e., collapsed, crushed) if `Filling` is `True`, `Auto indent` is `False`, and the insertion is exited with (Select). Use the Set command to set `Filling` to `False` before inserting in a table or list, or include a local margin command such as `^^M off` or `^^M off on`. (Shift)-(Select) exits the insertion mode without inserting and without margining.

If `Filling` is `False`, a beep is generated as you approach the end of the line, signaling you to press (Return) the same way a bell on a typewriter does. If you continue to insert text past the last visible column on your screen, the Editor accepts the characters and shows you that they are outside of the display area by placing an exclamation point (`!`) in the final column. To see these characters, complete the insertion by pressing (Select), position the cursor before the last word on the line and press (I) followed by (Return) to insert a carriage return.

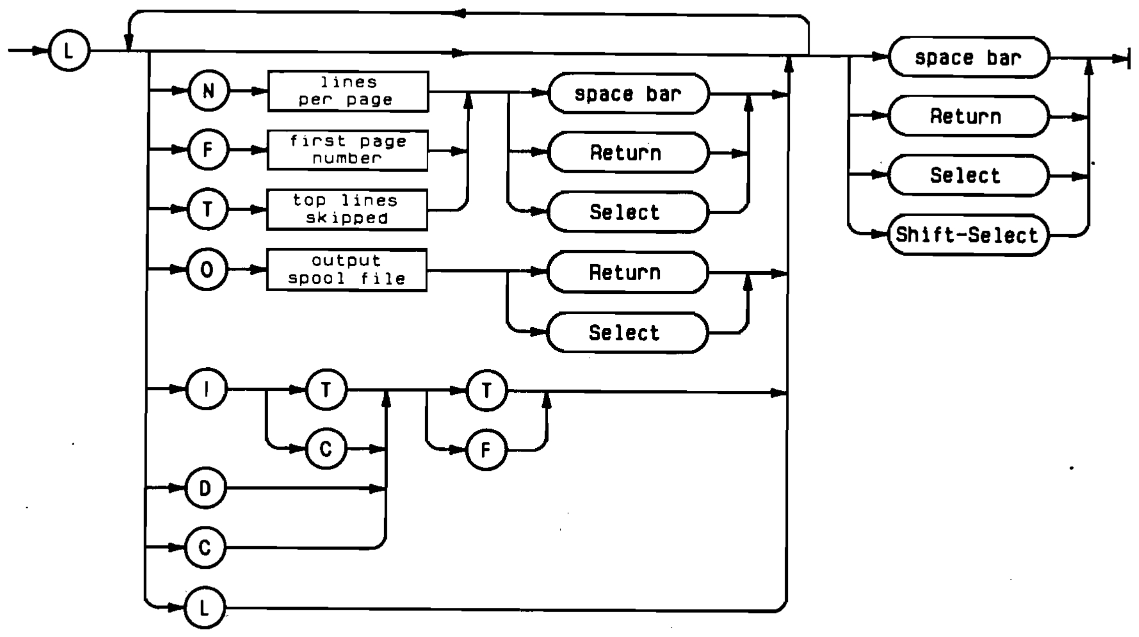
If `Auto indent` is `True`, pressing (Return) automatically places the cursor in the same starting column as the previous line. When `Auto indent` is `False`, the cursor is positioned according to either the `Left margin` or `Para margin` in the environment. If this is the first line of the paragraph, the cursor is placed in the column specified by `Para margin`. Otherwise, it is placed in the column specified by `Left margin`.

The entire insertion is stored in the copy buffer so you can copy the same text elsewhere if you wish. If Insert is exited with (Shift)-(Select) (regardless of the options set), all changes are aborted and the text and cursor appear as they did when the command was entered. Margining is not done.

The Insert command is one of the commands which sets the anchor (used by the Zap command) at the position where Insert was initiated. The anchor is set regardless of whether (Select) or (Shift)-(Select) is used to exit the command.

List

The List command will print the file currently in the Editor.



Item	Variable Type Parameter	Range Restrictions	Recommended Range
lines per page	integer constant	1 thru 9999	1 thru approximately 66
first page number	integer constant	0 thru 9999	–
top lines skipped	integer constant	0 thru lines per page minus one	–
output spool file	any valid ASCII file specifier, including volume name or pathname if necessary	≤ 72 characters	–

Semantics

Pressing L results in the following menu being displayed:

```
>List setup: {options}; <L lists>, <sel> or <sp> leaves

Number of lines per page:  60              Include Tag:        True
First page number:         1                  "    Commands:   False
Top lines skipped:         0
                                           Draw figures:       True
Printer type:              2934A           Continuous form:    True
Output spool file - PRINTER:
```

You may change the values of the various parameters by pressing the capitalized letter(s) of the option name, followed by the parameter value. On those parameters which only need a single character to complete the definition (such as `Draw figures`), the parameter entry will be completed as soon as you press the appropriate character. On those parameters which require more than a single character for complete definition (such as those requiring numbers), press (Return) to finish entry of the parameter. Pressing (Shift)-(Select) before completing a parameter entry restores the parameter to the value it had before you requested the change.

The parameters defined in this menu are part of the file's environment, and are stored with the file if the type is `.TEXT`. This means that when you load the file again, these parameters will be remembered and you won't have to define them again.

All list parameters default to the values listed in the *Customizing the Default Environment* section of the *Fully Utilizing HP TechWriter* chapter, or to values specified in the configuration file.

Number of lines per page
Select this option by pressing (N). This defines the number of lines total there are on a page. Some of them may be skipped at the beginning of a page, and some more at the end, but they are all counted when dealing with this parameter. This may be changed to whatever you wish, but it cannot be less than the `Top lines skipped` parameter, as this would result in nothing being printed on every page. Therefore, this condition is prohibited by the Editor.

First page number
Select this option by pressing (F). This parameter specifies the number of the first page that will be printed. This page number will be overridden by a document command which changes the page number, such as `^^P n=34`.

Top lines skipped
Select this option by pressing (T). This parameter defines how many lines at the top of each page are to be skipped before a printed line is output. Sometimes, if text is printed on the first line of a page, it looks too close to the top. In this case, skipping two or three lines may be more aesthetically-pleasing.

This value cannot be greater than the `Number of lines per page` value. If you attempt to specify an illegal value, you will be asked to redefine it.

Printer Type
Select this option by pressing (P). When you select this option, the following prompt appears:

```
**Legal printers:
    UNKNOWN, 2631G, 2671G, 2673A, 9876A,
    ThinkJet, 2932A, 2933A, 2934A, 82906A
```

Enter the type of printer you have. If you have a printer which is not mentioned above, then select the printer listed above that most resembles your printer in its operations, or select UNKNOWN. If your pictures aren't drawn (or are drawn incorrectly), set the Draw Figures parameter to False.

If the printer type is set to UNKNOWN, underlining will not be done.

Output spool file
Select this option by pressing O. This parameter defines where the file will go when being listed. If you have a local printer connected to your computer and you specify `PRINTER:` (the default), the listing will go to device 701, which means select code 7, device address 1.

If you are on an SRM system and wish to use that printer, you would probably change the definition of `Output spool file` to something like `/LP/Myfile.ASC`. Note that the file type must be `.ASC`.

Include Tag
Select this option by pressing I T and then specify T or F. If this is set `True`, each page will have a tag (footer) line. The contents of the tag is determined by the line following the most recent `^^T` document command. If there is no `^^T` document command in the file (or one hasn't been encountered by the time the bottom of an output page is reached) then the default as defined in the configuration file is used.

Include Commands
Select this option by pressing I C and then specify T or F. If this is set `True`, document commands will be printed, as well as error messages for each invalid document command. If `False`, the document commands will not be listed.

Draw figures
Select this option by pressing D and then specify T or F. If it is `True`, the figures referenced in your file will be printed; if it is `False`, they will not.

Continuous form
Select this option by pressing C. This parameter will almost always be `True`. "Continuous form" means that successive pages are connected together and feed automatically. Most printers normally print this way. An exception is the ThinkJet printer when you are using single-sheet operation. For single-sheet operation, which requires that you take the printed sheet out and insert a blank sheet before continuing printing, set `Continuous form` to `False`.

If `Continuous form` is false, the computer pauses at the top of every page and displays, "`Enter space when ready to continue.`"

Initiating the Listing

When you press L from within this menu, the listing starts. A message is displayed indicating the listing is being made:

```
>Listing ...
```

If you are listing to a local printer, it is obvious that the file is being listed, but if you are listing to an SRM printer, the listing is merely being sent to a spool file; the actual printing does not take place until you exit the L command by pressing Select or the space bar.

Multiple Copies
Multiple copies of your file may be printed by pressing L as many times as desired, once per copy. For example, if you want three copies of your file, press L L L. The keypresses go into the type-ahead buffer and each keystroke is processed when its turn comes. If there are too many Ls in the type-head buffer, you can edit it:

- CTRL-Backspace deletes the last (the rightmost) character from the type-ahead buffer, and
- CTRL-Clear line deletes all characters from the type-ahead buffer.

Printer Problems
If you attempt to list to a printer which is disconnected, powered down, or off-line, the printer eventually times out, and you get an error message which says:

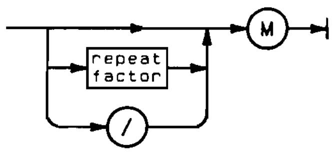
```
* Printer timeout: Fix or press <stop> to abort *
```

This message also appears if the printer goes off-line in the middle of printing, e.g., it may have run out of paper. The message appears after trying for approximately twelve seconds to get the printer to respond.

Stopping a Listing
If you want to stop a listing which is already in progress, press the Stop key. Control will be returned to the main Editor menu.

Margin

Margin formats all text in the next *n* paragraphs to fit the margins set in the environment or specified locally by a `^^M` command.



Item	Variable Parameter Type	Range Restrictions
repeat factor	integer numeric constant	1 thru 9999

Margin can only be executed in Document mode. If the Editor is not in Document mode when the Margin command is entered, the computer beeps and an error message is given:

```
>ERROR: Wrong environment <space> continues.
```

The Margin command provides a means of formatting paragraphs in your file. A paragraph is defined by the Editor to be *any* text delimited by any combination of blank lines, lines having the Command character as the first non-blank character in a line, or the beginning or end of a text file. See the Set command for details on the Command character.

Upon initiating Margin (by pressing `M`), the Editor takes all the text in the current paragraph (if the repeat factor is unspecified or 1), or the next *n* paragraphs (when the repeat factor is set greater than 1), and fits it within the paragraph's `Left margin`, `Right margin` and `Para margin` boundaries. The "next" *n* paragraphs may extend backward or forward, depending on the current direction specified in the prompt (`>` or `<`), and it always includes the paragraph in which the cursor currently resides.

There are three ways to define the margin boundaries for a paragraph:

- By using the global margins defined in the environment.
- By a margin document command immediately above the paragraph currently being margined.
- By a preceding "locking margin" command; that is, a document command of the form `^^M` *<margin parameters>* `ON` which has not yet been reset by a subsequent `^^M` document command.

After margining, the first line of the paragraph begins at the column specified by the `Para margin` (either local or global) setting and the rest of the text conforms to the `Left margin` and `Right margin` settings (either local or global). If a word would exceed the right margin it is "wrapped around" to the next line.

When a global or local parameter `Justify` is `True`, extra blanks are inserted between words to create a smooth right margin. The blanks are inserted from alternating ends of the lines of text to prevent the justified text from looking "denser" on one side than the other, as it would if the blanks were always inserted starting from the same edge. This means that if you are repeatedly margining a paragraph with an odd number of lines, the paragraph may look slightly different every other time.

Underline characters and escape sequences are not considered when margining. Thus, if you use underline characters, the lines containing them will extend past the right margin (this is especially noticeable when `Justify` is `True`). This is because when printing the text, the underline characters themselves are stripped out, and the underlining operation is merely turned on and off, which does not require extra character positions. Thus, the *printed copy* is justified as desired. A similar operation occurs for escape sequences.

Two blanks can follow the seven characters: `! . ; : ? " )`. All other blanks can be compressed into a single blank character. "Can be," rather than "are," because if a paragraph is margined when `Justify` is `True`, spaces may be added after other characters to make a smooth right margin, but they will be collapsed again if the paragraph is subsequently margined with `Justify False`. However, double spaces following the above seven characters will *not* be collapsed.

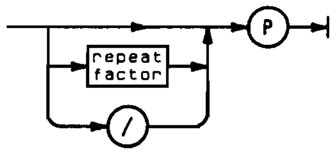
Note

If a table or list fits the definition of a paragraph, use of the Margin command **will** margin that text if Document mode is set. Exiting the Insert command with (Select) also uses the Margin routine, so be aware that these commands can potentially "collapse" a table or list.

The Margin command has no parameters except the repeat factor and its effects on a single paragraph cannot be aborted. Although the margin with a repeat factor can be aborted by pressing (Shift)-(Select), the paragraphs already margined cannot be reverted to their previous state unless you do it by hand. So when writing program text or tables, it is advised that Program mode be set, or the table be begun with `^^M off on`.

Page

Page moves the cursor one or more display pages (23 lines) in the current direction.



Item	Variable Parameter Type	Range Restrictions
repeat factor	integer numeric constant	1 thru 9999

Semantics

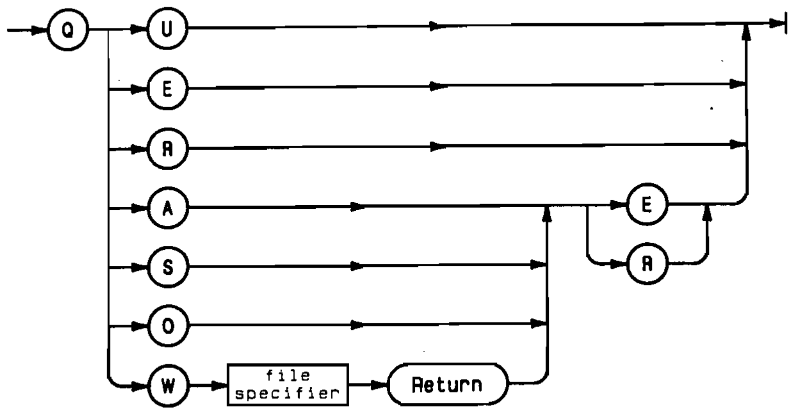
The Page command lets you move rapidly through a text file by repositioning the cursor one or more display pages (23 lines of text) forward (>) or backward (<) in a file. Page is executed by pressing P and its movement occurs relative to the position of the cursor. Page moves the cursor in the direction displayed by the Editor prompt when the command is entered.

Page is equivalent to using a repeat factor of 23 with ▲ or ▼ depending on the direction shown in the prompt. You can move *n* screen's distance by pressing *n* P, where *n* is a repeat factor between 0 and 9999. If $n=0$, the cursor merely moves to the beginning of the current line.

Select and Shift-Select are not available in the Page command. The command is immediately executed when P is pressed.

Quit

Quit leaves the Editor with various exit options.



Item	Variable Parameter Type	Range Restrictions
file specifier	literal	any valid file specifier

Semantics

The Quit prompt can take two forms. If the Editor knows what file name the current text came from, it will display:

```
>Quit:
     Update the workfile and leave
     Exit without updating
     Return to the editor without updating
     Write to a file name and return
     Archive file, write new file VOLUME:Fuel.TEXT
     Save as file new file VOLUME:Fuel.TEXT
     Overwrite as file VOLUME:Fuel.TEXT
```

If you entered the Editor with a workfile defined, or had specified no file, having started from scratch, the Editor's Quit prompt will look like this:

```
>Quit:
     Update the workfile and leave
     Exit without updating
     Return to the editor without updating
     Write to a file name and return
```

The Quit command allows you to exit the Editor and store your file on a mass storage medium. Quit is initiated by pressing (Q) from the Editor's prompt. Choose any of the options displayed by pressing the first letter (the capital letter) of the option.

Update the workfile and leave

If you press (U), the Editor will store the contents of your file under the name `WORK.TEXT` on the system volume. This workfile may or may not be associated with another file name (see the Get and Save commands in the chapter *Using the Filer* of this manual). `WORK.TEXT` is a standard name that many of the other subsystems in the Pascal Language System look for first. After writing the file, the system reports the file's size (in number of bytes) and displays the main command prompt.

Exit without updating

If you press (E) for Exit, having modified the file in any way (even global or List environment parameters), the Editor responds with:

```
E
Are you sure you want to exit without updating?
     Type Yes  to Exit Without Update
     Type No   to Return to Editor
```

If you press (Y) for yes, the Editor exits and your text in memory is destroyed. If you press (N) for no, you are returned to the Editor, and nothing is changed.

There is a valuable function for the the (E) option. If, while editing a file, you accidentally destroy a major portion of it, or realize your whole approach is wrong, or just want to start completely over, you can exit without updating, re-enter the Editor and specify the same file you started on before. This is possible because the Editor edits a *copy* of the file in RAM, not the file itself.

Pressing (E), not having changed anything in the file, causes the Editor to immediately exit.

Return to the editor without updating

Pressing (R) returns you to the Editor, with the cursor in the same place in the text as it was prior to pressing (Q). No file will have been made or updated. Typically, this is used to recover from accidentally pressing (Q).

One reason to deliberately Quit the Editor with the intention of returning with an (R) is just to look at the file name the Quit prompt shows you.

Write to a file name and return

If you press (W), the Editor will ask for a file name under which to store the text. This is the option you would select if no file name had ever been associated with this text, or you wanted the text to be stored with a different file name than the previously-specified name.

If you use the Write option and the file already exists, the Editor displays this prompt:

```
>Quit:
FILE.TEXT exists ...
  Rewrite then purge old
  Overwrite
  Purge old then rewrite
  Archive old then rewrite
  None of the above
```

`Rewrite then purge old`	An attempt is made to write the new file before purging the old.
`Overwrite`	This removes the original file and then attempts to write the new version in its place. On SRM units, duplicate links and passwords will be preserved. On a disc, the file may not fit if the new version is larger than the old.
`Purge old then rewrite`	This removes the original file and then attempts to write the new file in the biggest space on the disc. This alternative gives you the best chance that there will be room for the new file.
`Archive old then rewrite`	This archives the file (see below), and then stores the current text into a file whose name is displayed on the screen.
`None of the above`	This returns you to the Editor. You may Quit again and write the file with a different name.

Archive file, write new file

Pressing `A` for **Archive** will archive the file, and store the Editor's current contents under the displayed file name. The old file will have a commercial "at" sign ("@") appended to its name (there cannot be two files with the same name in the same volume). The Editor's current contents will be placed into a file under the name displayed on the screen.

For example, assume we loaded a file called `Fuel.TEXT`, modified it, and executed the Quit command. If `A` is pressed, the old file `Fuel.TEXT` would be renamed to `Fuel@.TEXT`. This doesn't affect the Editor's contents because, as mentioned before, the Editor doesn't edit the specified file, it edits a *copy* of the file. The Editor's contents are then placed in `Fuel.TEXT`. In this way, an original version of the file (prior to being edited) can be retained.

If there had already been a file named `Fuel@.TEXT`, it would have been purged *without notice* before the unedited version of `Fuel.TEXT` had been renamed to `Fuel@.TEXT`; that is, it will *not* make a file two generations old and call it `Fuel@@.TEXT`. If you want the file `Fuel@.TEXT` to be preserved as `Fuel@@.TEXT` then you must do the renaming explicitly with the Filer.

If your file name is already the maximum permissible length for the type of device it resides on, no "at" sign (`@`) can be appended to its name. In this case, the Editor will beep and give an error message. For example, assume the following:

- You want to store the file under the name `LargeFile`, which is nine characters long, allowing one character (a "`T`") to be suffixed to show file type of `.TEXT`.
- You are attempting to archive the file on the rightmost disc drive of a Model 36 computer, which allows ten-character file names.

The resultant file, `LargeFile@.TEXT`, would be encoded for the media as `LargeFile@T`. This is eleven characters long, however, and thus is an illegal file name for the internal floppies. The following error message would result:

```
A

Archive file error:  Bad file name
     Choose a different quit option.
```

Your response would be to store the file by selecting one of the other storing options.

Save as file new file
If you press (S), the edited version of the file (currently in the Editor) is stored under the same volume name and file name as the old file. This is done in such a way that the new file is stored under a temporary name, the old version is purged, and then the new version with the temporary name is renamed to the desired file name. In this way, if the store process aborts unexpectedly, the old file still exists; only *after* the storing process has successfully completed, is the old version removed.

If you try to Save a file and you get the message:

```
>ERROR: No room on vol <space> continues.
```

Press the space bar to continue. You could put in another mass storage medium with enough space, then Quit and Save it on the new disc. Alternatively, you can Quit and try to Overwrite the file if it has not gotten larger.

Overwrite as file
If you are on an SRM system, it is possible to have many people access the same file as if it were in their own directories. It is connected by what is called a *duplicate link*, and it prevents the necessity of either having a separate copy of a file in everyone's directory or requiring everyone to access the file through what may be a very long pathname.

When making an updated version of a file which has duplicate links pointing to it, use the Overwrite option, as it is the only one which will maintain the integrity of the duplicate link pointers. If you use any other option, the duplicate links will have to be redefined.

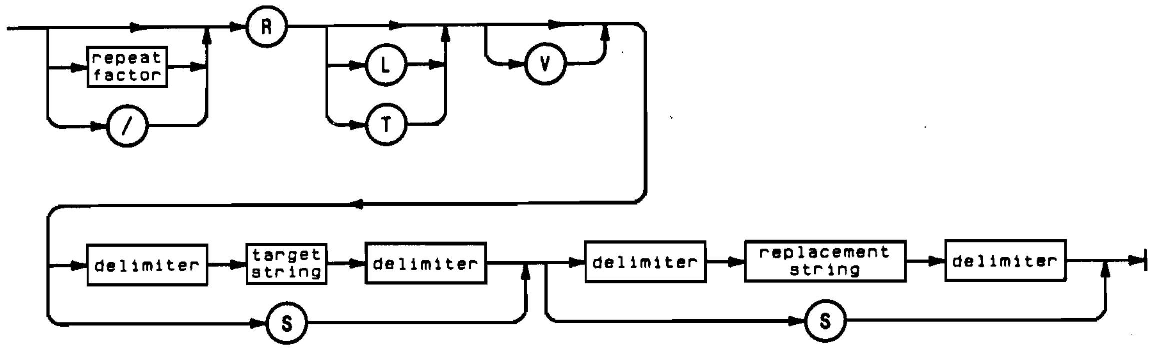
On a local disc, Overwrite may not work if the file has been enlarged. If this happens, you will get the following message:

```
>ERROR: file cannot be extended
Name of output file (<ret> to return) -->
```

Press (Return) to continue, Quit and Save again. The previous Overwrite removed the original file. Now the Save will try to save the file in the largest space on the mass storage medium. If this does not work, you must put the file on another mass storage medium. If duplicate links were defined, they will have been lost during the Save, so they will need to be redefined.

Replace

Replace does zero or more substitutions of a specified string with another string.



Item	Variable Parameter Type	Range Restrictions
repeat factor	integer numeric constant	1 thru 9999
delimiter	literal	any valid delimiter; must be used in matched pairs
target string	literal	1 thru 128 characters
replacement string	literal	1 thru 128 characters

Semantics

The Replace prompt:

```
>Repl[1]: L  V  <target><repl>=>
```

or

```
>Repl[1]: T  V  <target><repl>=>
```

The prompt displayed depends on whether the "Token" definition in the Editor's environment is set to `True` or `False`. If it is set to `True`, the first prompt is displayed; if it is `False`, the second is shown. These are explained below.

The Replace command allows you to replace one string with another in your text file. Replacements can be done to a single, all, or only certain occurrences of a string.

In its simplest form, the Replace command is executed by pressing `R` and specifying two strings – a target and replacement – each surrounded by delimiters. The target and replacement strings may be different sizes. Upon typing the final delimiter, the first occurrence of the target string is replaced by the replacement string and the cursor is positioned at the end of the substitution.

A **target string** (the one that you want replaced) must be supplied. A string is a contiguous series of non-control ASCII characters surrounded by delimiters. A delimiter signifies the beginning and end of a string and is one of the following characters:

```
! " # $ % _ ' ` + - * / ^ = . , : ; < > ( ) [ ] { } | \ ~ @ & ?
```

A **replacement string** (what you want the target string changed to) must also be supplied with delimiters. The replacement string may be an empty (null) string. For example, if you wanted to remove (replace with nothing) an occurrence of the word `the`, you could specify `/the///`. Notice that there is nothing between the final two delimiters, which specify the replacement string. Of course, you can use a repeat factor in this "deletion" operation.

Using a delimiter that appears within your string will not work. The Editor assumes when you press the character to be *in* your string (which you have already used as the beginning delimiter) that you have just ended the string by pressing the closing delimiter. Thus, delimiters must be in matched pairs, i.e., if you use "$" to signify the beginning of a string, you must use "$" to signify its end. The replacement string can have different delimiters than the target string and the two strings may be of different sizes. The maximum length of each string is 128 characters.

After pressing `R`, the prompt on your screen contains either an `L` or `T` for "literal" or "token" modes. Literal and token are interdependent; if one option is shown as available, the other is automatically the default. If `L` is shown in the prompt and you want to use the token search mode, then type in the two strings and their delimiters. The replacement takes place in the default mode, in this case, token. To do the same replacement in the literal mode, press `L` and type in both strings as before. The Replace command then searches for a literal form of the string.

A **literal string** is a string of characters either isolated or embedded in a word or line. A **token string** is one which is isolated by delimiters, which are any ASCII characters except numbers or alphabetic characters. Blanks are the most common delimiters in English text because they separate words.

Direction applies in the Replace command and is shown by the first character in the command's prompt. If the direction is forward (>), the replacement occurs between the cursor position and the end of the file; if backward (<), between the cursor and the beginning of the file.

Repeat factors are available for the Replace command but must be typed before the command is initiated (before `R` is pressed). A repeat factor causes that number of replacements to be made or tentatively made. If in Verify mode, having pressed `5` `R`, the Editor will ask you five times whether or not to replace the indicated string, regardless of how many you say "Yes" to. It will **not** keep going until you say "Yes" to five replacements.

If a repeat factor is not specified, the repeat factor defaults to 1. The slash (/) may also be used to change all occurrences of the specified string in the current direction. The repeat factor (or slash) is displayed in brackets ("[" and "]") in the command's prompt.

If a finite repeat factor is used, and at least one occurrence of the target string is found, but not as many as you specified, the Editor beeps a displays a message similar to the following:

```
>WARNING: Replaced 35, <space> continues.
```

Press the space bar to continue the editing session.

The **Verify option** lets you choose whether or not to make a particular replacement. The combination of a repeat factor with Verify allows you to replace only certain occurrences of a string in the file. For example, after pressing (2) (R) (V) and typing in the target and replacement strings, the Editor moves the cursor to the first occurrence of the target string and prompts:

```
>Repl[2]: R replaces, ' ' doesn't, <sh-sel> aborts
```

To confirm the replacement, press (R). To skip to the next replacement (if any), press the space bar. While using Verify, pressing (Shift)-(Select) aborts the operation and *retains* all replacements made up to that time.

If a finite repeat factor is used *with verify on*, and at least one occurrence of the target string is found, but not as many as you specified, the Editor beeps and displays a message similar to the following:

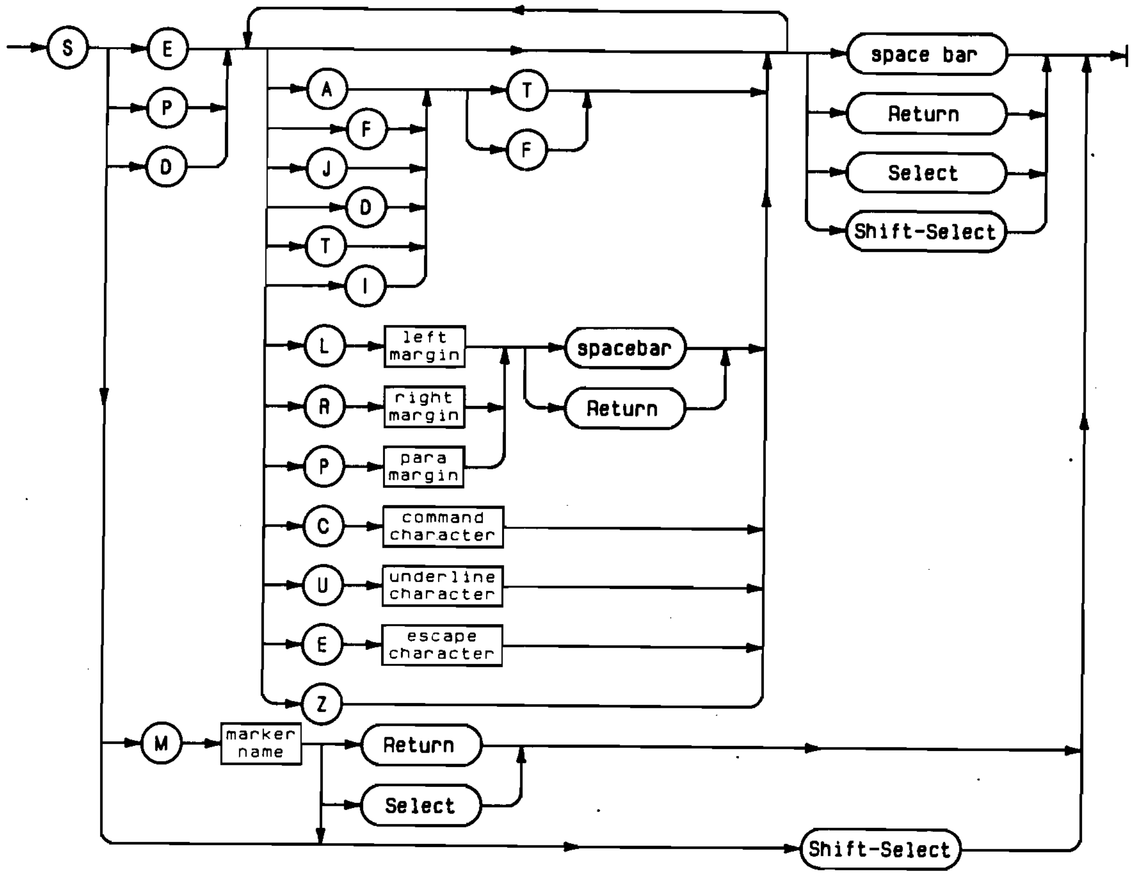
```
>WARNING: Verified 35, <space> continues.
```

Press the space bar to continue the editing session.

The **same** option is available with Replace and refers to either the most recent target string (used in a Find or Replace) or the most recent replacement string (used only in Replace). Which string it signifies (target or replacement) depends on where it is used in the Replace command. To use "same", simply press (S) in place of the delimited string. If you type (S) followed by a delimited string, the most recent target is replaced with the specified string. If you type a delimited string followed by (S), the specified target is replaced with the last replacement. Both strings may be specified by typing (S) (S). The current assignments of the "same" patterns can be seen by pressing (S) (E) (see the Set command for more details).

Set

Set defines markers and displays and alters the environment in which your text operations occur.



Item	Variable Parameter Type	Range Restrictions
left margin	integer numeric constant	0 thru right margin minus 1
right margin	integer numeric constant	1 thru 255
para margin	integer numeric constant	0 thru right margin minus 1
command character	literal	any valid ASCII character excluding CHR(0) thru CHR(32) and CHR(127); must be different than underline character and escape character
underline character	literal	any valid ASCII character excluding CHR(0) thru CHR(31) and CHR(127); must be different than command character and escape character
escape character	literal	any valid ASCII character excluding CHR(0) thru CHR(31), 0 thru 9, and CHR(127); must be different than command character and underline character
marker name	literal	1 to 8 ASCII characters excluding CHR(0) thru CHR(32), comma, and CHR(127)

Semantics

The Set prompt:

```
>Set:  Env Mark Prog Doc  [<sh-sel> to leave]
```

Set lets you define markers and various environment parameters. Markers are set by moving the cursor to where you want the marker, pressing S M (for Set Marker) and typing in a marker name followed by Return. A legal marker name is any sequence of up to eight characters. The characters which cannot be in a marker name are:

- Control characters (ASCII codes 0 through 31),
- Blanks (ASCII code 32),
- The "rubout" or "smudge" character (ASCII code 127), and
- Commas.

Any occurrences of these characters are deleted by the Editor.

Marker names are not case sensitive. The Editor converts all marker names to upper case letters so they can be typed using any desired combination of upper case and lower case letters.

No more than ten markers can be set in a file. If you attempt to set more than ten, the Editor displays the markers in a numbered list and prompts you for the number of the marker you wish to replace. Specify the marker to be replaced by typing its number (0 through 9), not its name. If you wish to delete *all* the currently defined markers, press Z while in environment-setting mode. Markers are used with the Jump and Copy commands and their names are shown in the environment display. The locations of the markers are not shown so the use of meaningful marker names is advised.

Pressing S E (for Set Environment) displays the current environment and allows you to change the environment's parameters. When entering the Editor with a new file, the default environment will be established as shown below, subject to modifications specified in the configuration file (see the section *Customizing the Default Environment* in the *Fully Utilizing HP TechWriter* chapter):

```
>Environment: {options} <sel> or <sp> leaves
    Auto indent   True                          Command ch     ^
    Filling       False                         Underline ch   \
    Justify       False                         Escape ch
    Left margin   1                             Draw figures   False
    Right margin  75                            Token def      False
    Para margin   5                             Ignore case    False
    Zap markers

    319 bytes used, 203774 available.
    System workfile
    Created:10-6-84  Used: 10-6-84  Type: new
```

Of course, the dates and the amount of memory available will probably be different, but the rest of the options will be as shown.

Patterns and Markers are only shown if they have been set. The heading near the bottom (which shows the file name now) displays a file name if the Editor is entered with a specified file. The entry labelled "`Type:`" specifies the current file type; if this is a new file that has never been put onto mass storage, it says `new`. If the file was retrieved from mass storage, the file type will be `TEXT`, `ASC`, or `DATA`. Whenever a file is saved as a `.TEXT` file on a mass storage medium, the current environment is saved with it and becomes the default environment when that file is used by the Editor.

The environment also displays how many bytes of memory have been used and how many are still available for use in the Editor. The total number of bytes available depends on the amount of memory in your machine.

To change a parameter in the environment, press the first letter in the parameter's name. The cursor is automatically positioned at the item to be changed and the new value must be typed. If the parameter needs a number (as in `Left margin`, `Right margin` and `Para margin`), then the number must be followed by pressing (Return) or the space bar. All other parameters accept a single character and return the cursor to the environment's prompt as soon as the character key is pressed. Those parameters whose values are true or false require you to press either (T) or (F). Any other key will elicit the response "`T or F`". When an invalid value is supplied, a correct value may be typed in its place, or you can press (Shift)-(Select) to restore the original value that the parameter had when you selected it.

Auto indent

Automatic indent is a true or false value which affects the Insert and Margin commands. When inserting text with this item set `True`, pressing (Return) automatically moves the cursor to the next line at the same starting column as the previous line. This indenting feature is useful when writing Pascal programs so it is set `True` for the Program (default) environment.

Filling

Filling is a true or false value which affects the Insert and Margin commands. When set `True`, filling causes automatic "word wrap" of text. When inserting, if a word extends beyond the current Right margin, it is wrapped around to the next line and no (Return) is necessary.

With `Filling` set `False`, the wrap around and margining functions are disabled. When approaching the end of a line, the Editor generates a "beep" to inform you that you need to press (Return) to go to the next line. If you type past the display area of the screen, an exclamation point (`!`) is shown in the last column. The text, though not visible, is maintained on the same line in the computer's memory.

Justify

Justify is a true or false value which specifies whether or not you want a *ragged right* edge (`Justify False`) or a smooth right edge (`Justify True`). Justification occurs only when a paragraph is margined (see the `Insert` and `Margin` commands for more information).

Left margin

The **Left margin** may be set to any integer between 0 and 254. The Left margin must be less than the Right margin setting or an error message is generated when you attempt to define that parameter:

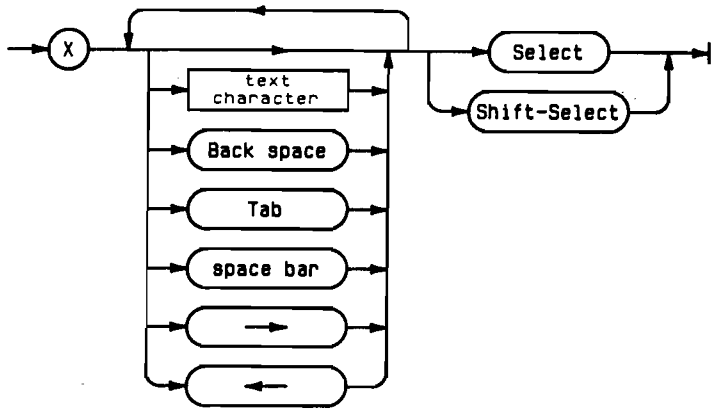
```
>ERROR: Improper margins <space> continues.
```

Right margin

The **Right margin** may be set to any integer between 1 and 255. Unless you have a particular reason for doing so (like making full use of a 132 column printer), it is not a good idea to set this margin beyond the right column display limits of your screen because the text will not be visible.

eXchange

eXchange replaces text character-for-character at the cursor position.



Item	Variable Parameter Type	Range Restrictions
text character	literal	any valid ASCII character excluding: CHR(0) thru CHR(31)

Semantics

The eXchange prompt:

```
Xchange: <text> <bs> [<sel> changes, <sh-sel> aborts]
```

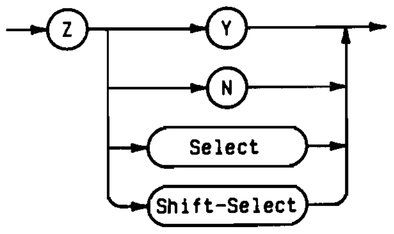
The eXchange command lets you exchange text character-for-character at the cursor position. eXchange operates only on the current line (i.e., the line where the cursor is located when the command is entered). The (Backspace), and (◀) move the cursor one space back and display the character that had been replaced. The (▶) causes the cursor to be moved to the right, leaving the original character.

Any non-control ASCII character can be used in eXchange. Carriage returns cannot be entered since the command is unable to cross line boundaries. Direction and repeat factors do not apply to the eXchange command. Backspacing past the point at which eXchange was entered is not allowed. No more than 128 characters can be exchanged in one operation of the eXchange command.

eXchange is initiated by pressing (X) and is exited by pressing (Select) or (Shift)-(Select). (Select) confirms the exchanges, returns the Editor prompt, and displays the cursor at its position when (Select) was pressed. (Shift)-(Select) returns the copy of the text file in the computer's memory to its state before eXchange was entered, displays the Editor prompt and shows the cursor at its position when eXchange was entered.

Zap

Zap deletes text and fills the copy buffer with the deleted text.



Semantics

The Zap command has only one option; confirmation of whether you really want to zap the characters. The Editor gives a message similar to the following when Z is pressed:

```
>WARNING: Want to Zap about nnn chars? (y/n)
```

where *nnn* is some number specifying the number of characters the Editor will delete if you press Y. The reason the word "about" is used in the message is that the Editor makes use of some control characters that you cannot see. These control characters add, on the average, three characters to every visible line of text. Therefore, the character count in a Zap confirmation message may appear to be off by three characters per line.

Zap deletes all text between the "anchor" and the current cursor position and stores it in the copy buffer. The anchor is located at the position in the text where the most recent Find, Adjust, Insert or Replace command (you can remember them by noting the initial letters: "FAIR") was initiated. You can confirm the position of the anchor with the Equals command, which moves the cursor to the anchor.

If the cursor position *is* the anchor position, that is, you are telling the Editor to zap zero characters, the Z keypress is ignored.

If the Copy buffer is not large enough to store the deletion, a prompt asks if you wish to go ahead and Zap the text anyway. Use the Set environment command to see how much memory is available; the copy buffer shares this memory with that used to hold the text file in memory.

Recovery of the deleted text is achieved with the Copy (from buffer) command. Zap can thus be used to move large chunks of text from one location to another within a file.

Note

The effects of Zap can be surprising since the anchor position is set by four different and commonly used commands (listed above). Therefore, it is a good practice to check the location of the anchor (using the Equals command) before executing a Zap.

Lister Reference

Chapter 10

The HP TechWriter Lister program allows you to produce listings of documents which are composed of multiple files. The listing produced may either be in the form of one large document or a series of individual files.

Up to fifty file specifiers can be entered for listing, as well as several environment parameters. These file specifiers and environmental parameters can be stored on a file and recalled later. At any time except during a listing, file specifiers can be viewed, modified, added, or deleted, and listing environment parameters can be modified. The Lister can also produce a table of contents from commands embedded in the files.

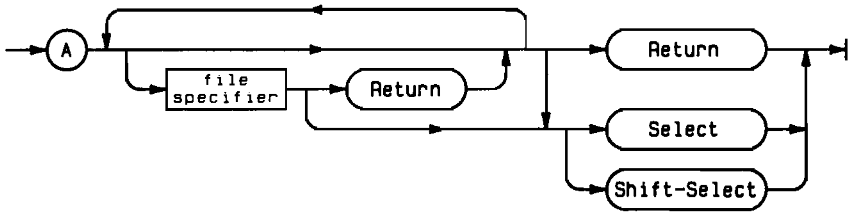
Instructions for running the Lister program are contained in the *Lister Tutorial* section of the *Getting Started* chapter.

This chapter contains a detailed description of the syntax and semantics of each HP TechWriter Lister command presented in alphabetical order.

The term "file pool" is used in this chapter to refer to the list of file specifiers you have entered for listing.

Add files

The Add files command allows you to add file specifiers to the end of the file pool.



Item	Variable Parameter Type	Range Restrictions
file specifier	literal; a file name preceded optionally by a volume specifier	no. of characters ≤ 72

The Add files command places the cursor after the last in the file pool, then prompts you for a file specifier. The file specifier should include any pathnames or volume names needed to locate the file. File specifiers may not be longer than 72 characters. The Lister automatically appends `.TEXT` to the end of any file specifier which does not end in a period.

There are three slightly different ways to stop Adding files. All cause the main Lister prompt to be re-displayed. You can:

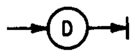
- Press (Return) on an empty line.
- Press (Select) immediately after a file specifier that you want added to the file pool.
- Press (Shift)-(Select) immediately after a file specifier that you don't want added to the file pool; the file on this line will *not* be added to the file pool.

If fifty file specifiers already exist, no more can be added. The following message will appear if you try to add more files:

```
No more room for file names.  <space> continues.
```

Delete

The Delete command deletes the file specifier currently indicated by the cursor.

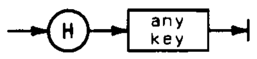


Semantics

The Delete command deletes the file specifier currently pointed to by the cursor. All subsequent file specifiers are moved up one place.

Help

The Help command fills the screen with text that briefly describes the commands and their operations.

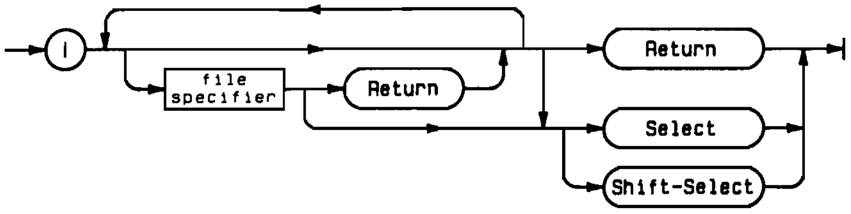


Semantics

The Help command erases the display of the file pool from the screen and replaces it with text that describes the commands available from the Lister. Pressing any alphanumeric key erases the Help text and returns the file pool display to the screen.

Insert

The Insert command allows you to insert file specifiers in the file pool.



Item	Variable Parameter Type	Range Restrictions
file specifier	literal; a file name preceded optionally by a volume specifier	no. of characters ≤ 72

Semantics

The Insert command erases the file specifier indicated by the cursor and all subsequent ones from the screen, and then prompts for a new file specifier. After inserting zero or more file specifiers, press (Return) or (Select). The erased file specifiers will be renumbered and reprinted on the screen.

The file specifiers should include any pathnames or volume names needed to locate the files.

There are three slightly different ways to stop Inserting files. All cause the main Lister prompt to be re-displayed. You can:

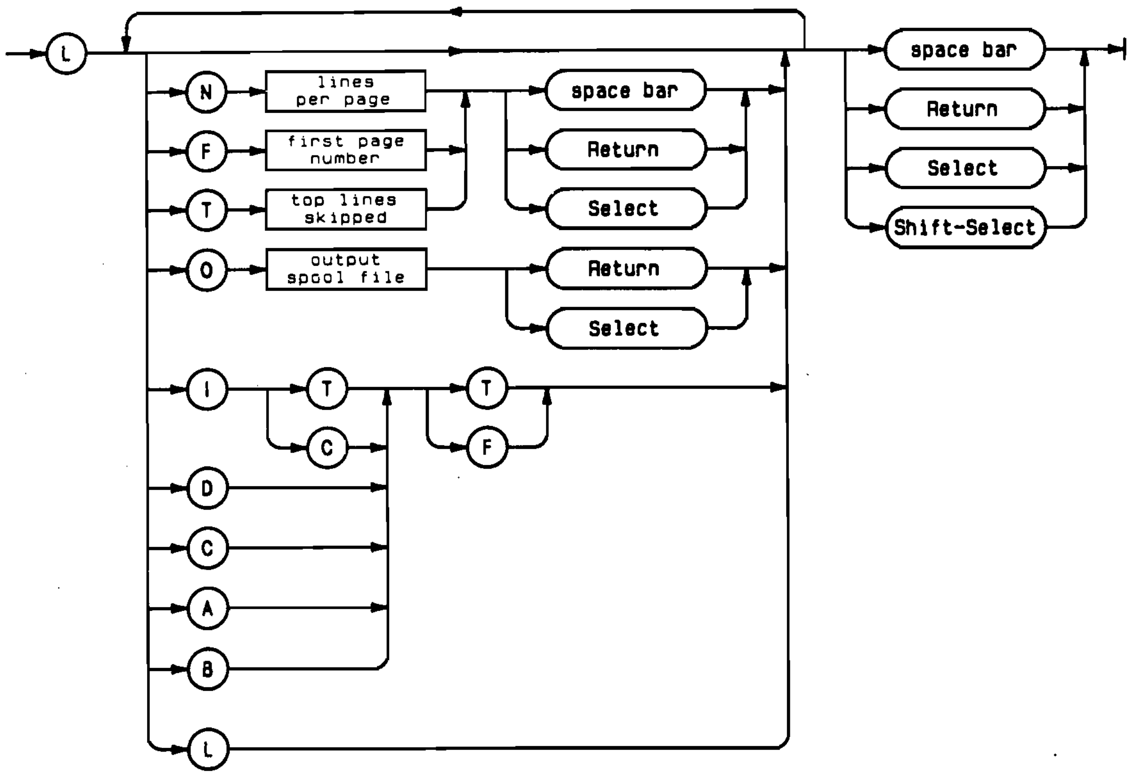
- Press (Return) on an empty line.
- Press (Select) immediately after a file specifier that you want inserted in the file pool.
- Press (Shift)-(Select) immediately after a file specifier that you don't want inserted to the file pool; the file on this line will *not* be put in the file pool.

If fifty file specifiers are already defined, no more can be inserted. The following message will appear if you try to insert more:

```
No more room for file names.  <space> continues.
```

List

The List command will print the files currently in the file pool.



Item	Variable Parameter Type	Range Restrictions	Recommended Range
lines per page	integer constant	1 thru 9999	1 thru approximately 66
first page number	integer constant	0 thru 9999	–
top lines skipped	integer constant	0 thru lines per page minus 1	–
output spool file	any valid ASCII file specifier, including volume name or pathname if necessary	≤ 72 characters	–

Semantics

When you have defined all the files you want listed, press L to enter the List environment menu.

```
List environment:  {options};  L Lists;  <sel> or <sp> leaves

Number of lines per page: 60        Include Tag:             True
First page number:        1            "    Commands:        False
Top lines skipped:        0         tAble of contents:       False
Begin new page/reset page           Draw figures:            True
number for each file:     False     Continuous form:         True
Printer type:             2934A
Output spool file - PRINTER:
```

You may change the values of the various parameters by pressing the capitalized letter(s) of the option name, followed by the parameter value. On those parameters which only need a single character to complete the definition (such as `Draw figures`), the parameter entry will be completed as soon as you press the appropriate character. On those parameters which require more than a single character for complete definition (such as those requiring numbers), press Return to finish entry of the parameter. Pressing Shift-Select before completing a parameter entry restores the parameter to the value it had before you requested the change.

The parameters defined in this menu are stored in the list file. This means that when you use this list file again, these parameters will be remembered and you won't have to define them again.

All list parameters default to the values specified in the configuration file if they are there. If they are not specified in the configuration file, the default values specified in the *Customizing the Default Environment* section of the *Fully Utilizing HP TechWriter* chapter will be used.

Listing Parameters

Number of lines per page

Select this option by pressing N. This defines the number of lines total there are on a page. Some of them may be skipped at the beginning of a page, and some more at the end, but they are all counted when dealing with this parameter. This may be changed to whatever you wish, but it cannot be less than the `Top lines skipped` parameter, as this would result in nothing being printed on every page. Therefore, this condition is prohibited by the Lister.

First page number

Select this option by pressing F. This parameter specifies the number of the first page that will be printed. This page number will be overridden by a document command which changes the page number, such as `^^P n=34`.

Top lines skipped
Select this option by pressing (T). This parameter defines how many lines at the top of each page are to be skipped before a printed line is output. Sometimes if text is printed on the first line of a page it looks too close to the top. In this case, skipping two or three lines may be more aesthetically pleasing.

This value cannot be greater than the `Number of lines per page` value. If you attempt to specify an illegal value, you will be asked to redefine it.

Begin new page/reset page number for each file
Select this option by pressing (B). This parameter allows you to specify whether you want a a new page started and page numbering reset at the beginning of every file. If it is set `False`, the files are considered one continuous document.

Printer Type
Select this option by pressing (P). When you select this option, the following prompt appears:

```
**Legal printers:
    UNKNOWN, 2631G, 2671G, 2673A, 9876A,
    ThinkJet, 2932A, 2933A, 2934A, 82906A
```

Enter the type of printer you have. If you have a printer which is not mentioned above, then select the printer listed above that most resembles your printer in its operations, or select UNKNOWN. If your pictures aren't drawn (or are drawn incorrectly), set the Draw Figures parameter to False.

If the printer type is set to UNKNOWN, underlining will not be done.

Output spool file
Select this option by pressing (O). This parameter defines device or file name for listing output. If you have a local printer connected to your computer and you specify `PRINTER:` (the default), the listing will go to device 701, which means select code 7, device address 1.

Include Tag
Select this option by pressing (I) (T) and then specify (T) or (F). If this is set `True`, each page will have a tag (footer) line. The contents of the tag is determined by the line following the most recent `^^T` document command. If there is no `^^T` document command in the file then the default as defined in the configuration file is used.

Include Commands
Select this option by pressing (I) (C) and then specify (T) or (F). If this is set `True`, the document commands will be printed. If it is set `False`, the document commands will not be listed.

tAble of contents
Select this option by pressing (A) and then specify (T) or (F). If this is set `True`, a table of contents will be printed using all `^^C` commands that were found as the Lister read through the file pool. If it is set `False`, a table of contents will not be generated. The table of contents features are discussed in the *Noting Table of Contents Entries* section in the *Fully Utilizing HP TechWriter* chapter.

Draw figures
Select this option by pressing D and then specify T or F. If it is `True`, the figures referenced in your file will be printed; if it is `False`, they will not.

Continuous form
Select this option by pressing C. This parameter will almost always be `True`. "Continuous form" means that successive pages are connected together and feed automatically. Most printers normally print this way. An exception is the ThinkJet printer when you are using single-sheet operation. For single-sheet operation, which requires that you take the printed sheet out and insert a blank sheet before continuing printing, set `Continuous form` to `False`.

If `Continuous form` is false, the computer pauses at the top of every page and displays, "`Enter space when ready to continue.`"

Initiating the Listing

Pressing L while inside the list menu starts the listing. The first thing that happens is that all the specified files are sought, and if any cannot be found, a message to that effect will be displayed. If none of the files are found, the following prompt is displayed:

```
ERROR: No files found <space> continues.
```

If some, but not all the files are found, the following prompt is displayed:

```
Missing files.  Produce the listing anyway? (y/n)
```

If you press Y, the listing will be made as if you hadn't specified the missing files' names. If you press N, you will be returned to the main Lister menu.

Multiple Copies
Multiple copies of your file may be printed by pressing L as many times as desired, once per copy. For example, if you want three copies of your file, press L L L. The keypresses go into the type-ahead buffer and each keystroke is processed when its turn comes. If there are too many Ls in the type-head buffer, you can edit it:

- CTRL-Backspace deletes the last (the rightmost) character from the type-ahead buffer, and
- CTRL-Clear line deletes all characters from the type-ahead buffer.

Printer Problems

If you attempt to list to a printer which is disconnected, powered down, or off-line, the printer eventually times out, and you get an error message which says:

```
* Printer timeout: Fix or press <stop> to abort *
```

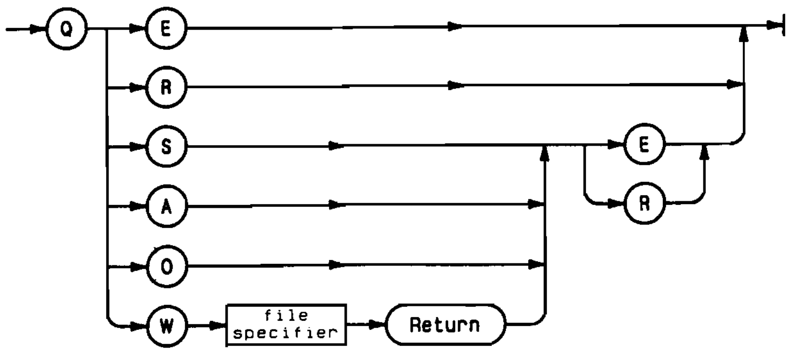
This message also appears if the printer goes off-line in the middle of printing, e.g., it may have run out of paper. The message appears after trying for approximately twelve seconds to get the printer to respond.

Stopping a Listing

If you want to stop a listing which is already in progress, press Stop. Control will be returned to the main Lister menu.

Quit

The Quit command exits the Lister, offering file-storing options.



Item	Variable Parameter Type	Range Restrictions
file specifier	literal; a file name preceded optionally by a volume specifier	any valid file specifier

Semantics

The Quit prompt can take two forms. If the Lister knows what file name the current data came from, it will display:

```
>Quit:
     Exit without updating
     Return to the lister without updating
     Write to a file name and return
     Archive file, write new file VOLUME:ListFile.LS
     Save as file new file VOLUME:ListFile.LS
     Overwrite as file VOLUME:ListFile.LS
```

If you entered the Lister having specified no file (having started from scratch), the Lister's Quit prompt will look like this:

```
>Quit:
     Exit without updating
     Return to the lister without updating
     Write to a file name and return
```

The Quit command allows you to exit the Lister and store your listing data on a mass storage medium. Choose any of the options displayed by pressing the first letter (the capital letter) of the option.

The Lister appends `.LS` to the file specifier you type, unless you end your file specifier with a period.

Exit without updating
If you press E for Exit, having modified the file pool or List environment parameters in any way, the Lister responds with:

```
Are you sure you want to exit without updating?
     Type Yes  to Exit Without Update
     Type No   to Return to Lister
```

If you press Y for yes, the Lister exits and your file pool and listing parameters in memory are destroyed. If you press N for no, you are returned to the Lister, and nothing is changed.

Pressing E, not having changed anything in the file, causes the Lister to immediately exit.

Return to the lister without updating
Pressing R returns you to the Lister, with the cursor in the same place in the file pool as it was prior to pressing Q. No file will have been made or updated. Typically, this is used to recover from accidentally pressing Q when trying to press A for Add files, or something similar.

One reason to deliberately Quit the Lister with the intention of returning with an R is to look at the file specifier the Quit prompt shows you.

Write to a file name and return
If you press W, the Lister will ask for a file name under which to store the file. This is the option you would select if no file specifier had ever been associated with this file, or you wanted the file to be stored under a different name than the previously-specified name.

If you use the Write option and the file already exists, the Lister displays a prompt similar to this:

```
>Quit:
FILE.LS exists ...
  Rewrite then purge old
  Overwrite
  Purge old then rewrite
  Archive old then rewrite
  None of the above
```

`Rewrite then purge old`	An attempt is made to write the new file before purging the old.
`Overwrite`	This removes the original file and then attempts to write the new version in its place. On SRM units, duplicate links and passwords will be preserved. On a disc, the file may not fit if the new version is larger than the old.

`Purge old then rewrite`	This removes the original file and then attempts to write the new file in the biggest space on the disc. This alternative gives you the best chance that there will be room for the new file.
`Archive old then rewrite`	This archives the file (see below), and then stores the current text into a file whose name is displayed on the screen.
`None of the above`	This returns you to the Lister.

Archive file, write new file
Pressing (A) for **Archive** will archive the file, and store the Lister's current contents under the displayed file name. The old file will have an "@" appended to its name (there cannot be two files with the same name in the same volume). The Lister's current contents will be placed into a file with the name displayed on the screen.

If an archive file already exists for this file, the existing archive file will be purged *without notice* before the new one is made.

Save as file new file
If you press (S), the version of the file currently in the Lister is stored under the same volume name and file name as the old file. This is done in such a way that the new file is stored under a temporary name, the old version is purged, and then the new version with the temporary name is renamed to the desired file name. In this way, if the store process aborts unexpectedly, the old file still exists; only *after* the storing process has successfully completed, is the old version removed.

If you try to Save a file and you get the message:

```
>ERROR: No room on vol <space> continues.
```

press the space bar to continue. You could put in another mass storage medium with enough space, then Quit and Write the file on the new disc. Alternatively, if the file has not gotten larger, you can Quit and try to Overwrite the file.

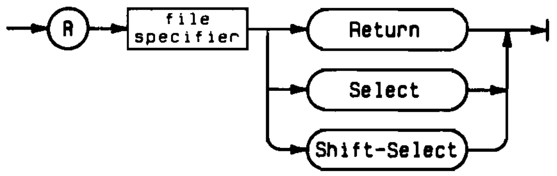
Overwrite as file
Overwrite may not work if the file has been enlarged. If this happens, you will get the following message:

```
>ERROR: file cannot be extended
Name of output file (<ret> to return) -->
```

Press (Return) to continue, Quit and Save again. The previous Overwrite removed the original file. Now the Save will try to save the file in the largest space on the mass storage medium. If this does not work, you must put the file on another mass storage medium. If duplicate links were defined, they will have been lost during the Save, so they will need to be redefined.

Replace

The Replace command allows you to replace an existing file specifier with a new one.



Item	Variable Parameter Type	Range Restrictions
file specifier	literal; a file name preceded optionally by a volume specifier	no. of characters ≤ 72

Semantics

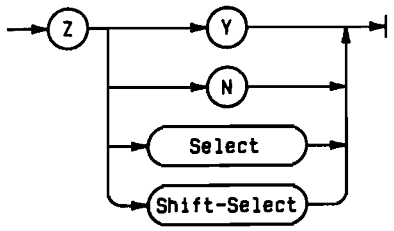
The Replace command allows you to replace the file specifier currently pointed to by the cursor with a new file specifier.

You can enter no more than one file specifier each time you invoke the Replace command. There are two ways that you can complete the command. You can:

- Press Return or Select on an empty line or Shift-Select at any time to cancel the Replace command and leave the existing file as is.
- Press Return or Select immediately after a file specifier to replace the original file with the new specifier.

Zap

The Zap command clears all file specifiers but leaves the List parameters unchanged.



Semantics

When you press Z, the computer beeps and says:

```
WARNING: Want to Zap n files? (y/n)
```

where *n* is the current number of files in the file pool. If you press Y, all file specifiers will be erased, but the listing parameters will remain unchanged. If you press anything else, the Lister prompt returns, and none of the files have been deleted.

If you have *n* files, pressing Z is the same as moving the cursor to the first file and pressing D *n* times.

Picture Processor Reference

Chapter 11

The Picture Processor converts plot files produced by graphics editors into picture files that the Editor and Lister can use for illustrating documents. You can get a list of the graphics programs that work with HP TechWriter from your local Hewlett-Packard Sales Representative. Appendix C, *Making Your Own Pictures*, discusses some of the options available to you.

The Picture Processor program is called `PICPROC` and was delivered on your `LIST:` disc.

Instructions for running the Picture Processor are contained in the *Picture Processor Tutorial* section at the end of the *Getting Started* chapter.

This chapter contains more information about the Picture Processor and about the plot file commands it understands.

Running the Program

After you start the Picture Processor program, it displays:

```
HP TechWriter PICTURE PROCESSOR      [Rev 1.0]  1-Jul-84

Copyright 1983 Hewlett-Packard Company,
         All rights reserved.

Plot file name <ret> (just <ret> to quit):
```

Type the exact name of the plot file in answer to this first question, including any volume names, device numbers or pathnames necessary to locate the file. The program will respond with another question:

```
New file name <ret> (just <ret> to quit):
```

Type the exact name of the file to be produced, including any volume names, device numbers or pathnames necessary to define the location of the file. If a file already exists with the name you specified for "New file name", the program responds:

```
Output file already exists. Do you want to
overwrite it anyway? (Y/N, <ret>)
```

As processing of the file proceeds, a row of periods will be printed one at a time – the longer the file. the more periods will be printed. When processing is complete, the screen will display:

```
Picture processing of file <plot file name> is complete.
Processed file <new file name> is available for use.
```

You may enter another file name for processing, or press **Return** with no file name to exit the program.

Exceptional Conditions

- If the file name you enter for the plot file does not exist or is a file which has already been processed, the program will give you a message to that effect and return to the first question.
- If the file name you entered for the new file name already exists, the program will ask you if you want to purge it. If so, the file will be purged, the new one created, and the program will continue processing. If you do not want to purge the file, the program will return to the second question.
- If the file name you entered for the new file name is the same as the plot file name, the program will return to the second question.
- If any errors occur, an error message will be displayed along with the following message:

  ```
  No processed file has been produced.
  ```

  The error most likely to occur is the Picture Processor running out of room on the disc on which the "new file" is being built.
- If (Stop) is pressed during processing, the following message will be displayed:

  ```
  Stopped by user request.
  ```

  In this case, no processed file will be produced and the program will be terminated.

Supported HPGL Commands

Plot files are written in *Hewlett-Packard Graphics Language* (HPGL). The Picture Processor translates HPGL commands into a binary form which can be quickly read by the HP TechWriter Editor and Lister.

HPGL commands are series of ASCII characters. Commands must be separated by line endings or semicolons. The Picture Processor requires that lines of commands in the input file contain no more than 255 characters. The Picture Processor also requires that all letters in the commands be upper case.

The HPGL commands which the Picture Processor translates are these:

`LT`	Line Type. Select a line type for subsequent lines. Line types include solid lines, dotted lines, dashed lines, etc. Line type has two optional parameters. It may have an integer type with values between −7 and 7. If it has a type specified, a floating point length may also be specified (this length is not used by HP TechWriter). Sample line type commands are: `LT`, `LT1`, and `LT1,,5`.
`PA`	Plot Absolute. Move the pen to the specified absolute coordinates. If the pen is up, the pen is merely moved; if the pen is down, a line is drawn. The Plot Absolute command must have at least one pair of integer parameters, and may have multiple pairs of integer parameters. The first integer in each pair is assumed to be an `X` value, and the second is assumed to be a `Y` value. Sample Plot Absolute commands are: `PA10,10` and `PA10,10,10,0,0,0,0,10,10,10`
`PD`	Pen Down. Lower the pen onto the plotting surface. No X or Y motion is done. This command has no parameters. A sample Pen Down Command is: `PD`.
`PU`	Pen Up. Raise the pen from the plotting surface. No X or Y motion is done. This command has no parameters. A sample Pen Up Command is: `PU`.
`SP`	Select Pen. The Select Pen command has one optional parameter, the pen number, which is an integer. Pen numbers are ignored in HP TechWriter except that plot commands following a pen number of zero ("no pen") are ignored until the next pen command.

The following are examples of legal commands:

```
LT
PU;
PD;PA10,10;PA10,0,0,0;
PA0,10;LT5;PA10,10
```

Filer Reference

Chapter 12

Filer Commands

This chapter contains a brief overview of the HP TechWriter Filer commands and a detailed description of the syntax and semantics of each command presented in alphabetical order.

Filer Command Summary

File Maintenance Commands

Change	Change the name of a file, set of files, or volume.
Filecopy	Copy a file, set of files, or a volume to a specified destination.
Make	Create a file on a volume.
Remove	Remove a directory entry or a set of directory entries.
Translate	Translate text files of types TEXT, ASCII, and DATA to other text file representations or to un-blocked volumes.

Volume Maintenance Commands

Bad sectors	Scan a volume and search for unreliable (bad) storage areas.
Extended directory	List directory information about a specified volume or set of files.
Krunch	Consolidate all unused space on a volume in a single area by packing the existing files together.
List Directory	List directory information about a specified volume or set of files.
Prefix	Specify a new default volume.
Volumes	List the volume currently on line.
Zero	Create an empty directory on the specified volume.

Workfile-Related Commands

Get	Specify a file as the workfile.
New	Specify that no file is the current workfile.
Save	Save the current workfile(s) with the specified name.
What	List the name and current state (saved or not saved) of the workfile(s).

Exit Commands

Quit	Make an orderly exit from the filer.
Stop	Pressing the (Stop) key unconditionally exits the Filer Subsystem. The current I/O operation is completed before exiting.

SRM-Related Commands

Note

The following commands appear in the HP TechWriter Filer prompt, but are used only if you have a Shared Resource Management system. These commands will not be described in this chapter. Please consult the *Pascal 3.0 Workstation System* manual for more information about Filer commands for the SRM.

Access	Change the access rights (passwords) on a SRM file or directory.
Duplicate link	Duplicate links to a file or set of files on the SRM.
Make	Create a SRM directory.
Udir	Set the default SRM unit directory.

Command Syntax and Semantics

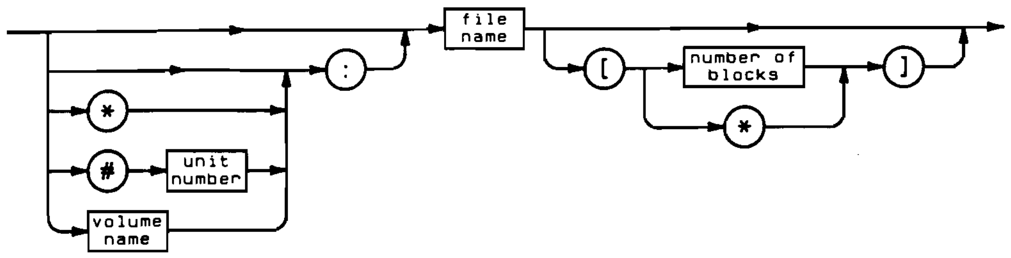
The Filer commands are presented in alphabetical order. Each command's explanation includes: the command's name, a brief functional description, a diagram showing its legal syntax, the command's prompt (if any) and a discussion of how to use each command. Each command's options are also covered.

Alphabetical List of Filer Commands

Bad sectors
Change
Extended directory
Filecopy
Get
Krunch
List directory
Make
New
Prefix
Quit
Remove
Save
Translate
Volumes
What
Zero

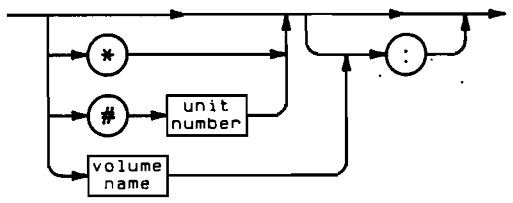
Several of the syntax diagrams in this chapter include the "volume specification" and the "file specification". The "volume specification" is the syntax for commands that operate on volumes. The "file specification" is the syntax for commands that operate on files. Volume specifications don't need the ":" except when a literal volume name is given. Then the name must end with a ":" to distinguish it from a file name. If no volume specifier is given, the default volume is assumed.

File Specification



Item	Variable Parameter Type	Range Restrictions
unit number	integer; corresponding to an entry in the unit table	1 thru 50
volume name	literal	any legal volume name
file name	literal	any legal file name
number of blocks	integer	any legal number of blocks

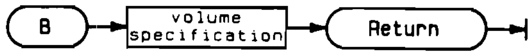
Volume Specification



Item	Variable Parameter Type	Range Restrictions
unit number	integer; corresponding to an entry in the unit table	1 thru 50
volume name	literal	any legal volume name

Bad sector

The Bad sector command scans a mass storage medium for errors.



Item	Variable Parameter Type	Range Restrictions
volume specification	literal	(See the beginning of this chapter)

The Bad sector prompt:

```
Bad sector scan of what directory ?
```

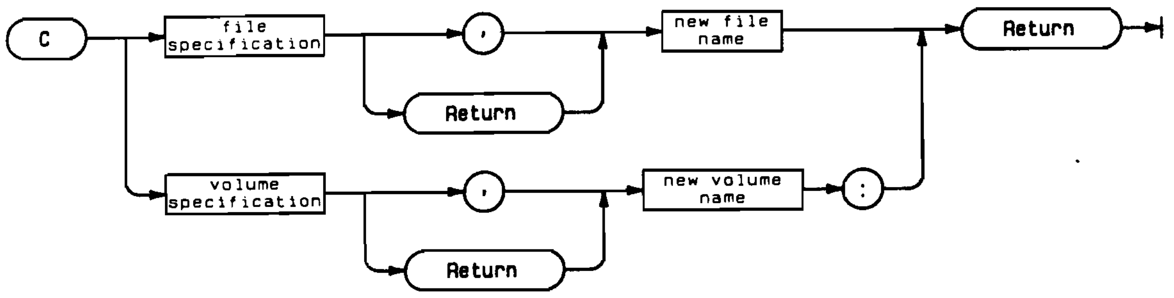
The Bad sector command allows you to check a mass storage medium to find out if each block (sector) is readable. A flexible disc may become unreadable after damage or excessive wear.

Press B to initiate the command and answer the prompt with a volume specification. The Filer then displays a message indicating that it is scanning the volume from block 0 to the end of the volume. The Filer does a read operation on each sector and does a CRC error check on the results. If the CRC results are normal, that sector is considered to be good; if not, the Filer lists the sector number.

If you find a bad sector in a file, you may wish to use the Filer to change the file type (suffix) to `.BAD`. The BAD file will not be moved in a Krunch operation. A large number of bad sectors indicates a worn-out medium. The medium should only be used if you are willing to risk losing information on that volume.

Change

The Change command lets you rename files and volumes.



Item	Variable Parameter Type	Range Restrictions
file specification	literal	(See the beginning of this chapter)
volume specification	literal	(See the beginning of this chapter)
new file name	literal	any valid file name
new volume name	literal	(See the beginning of this chapter)

Semantics

The Change prompt:

```
Change what file?
```

The Change command requires two specifications: the original volume or file specification and the new one. The two specifications can be separated by either a comma or a carriage return. If you are changing the name of a volume, any legal volume ID can be used for both specifications.

To change the name of a file, use its volume ID in the first specification and only the new file name in the second specification. The Filer assumes that the file whose name you are changing resides on the volume identified in the first specification. After the Filer has finished changing the name and updating the directory it reports the name changes it has made.

Wildcards (the `=` and `?` characters) may be used in the Change command. If a wildcard is used in the first specification, it must also be used in the second one. The subset string that is replaced by the wildcard in the second specification (the new name) is the same as the string it stands for in the first specification.

Suppose you have a volume named `STUFF:` with the following files:

```
WHATISIT.TEXT
WHOISIT.TEXT
WHYISIT.TEXT
```

Specifying `STUFF:WH=TEXT, FO=FA` in response to the Change prompt results in the following messages being reported by the Filer:

```
STUFF:WHATISIT.TEXT changed to FOATISIT.FA
STUFF:WHOISIT.TEXT changed to FOOISIT.FA
STUFF:WHYISIT.TEXT changed to FOYISIT.FA
```

Here is another example using the original files shown above on the `STUFF:` volume. Specifying `STUFF:WH=.TEXT, =` results in:

```
STUFF:WHATISIT.TEXT changed to  ATISIT
STUFF:WHOISIT.TEXT  changed to OISIT
STUFF:WHYISIT.TEXT  changed to YISIT
```

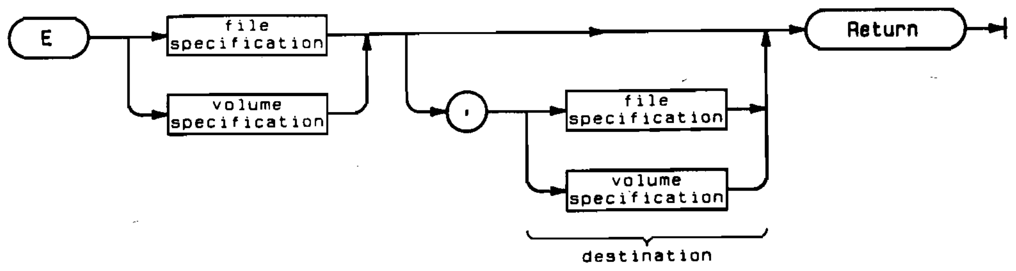
You may wish to create some empty files using the Make command and experiment with them before using wildcards extensively. Until you get used to them, the effects of wildcards are not always obvious.

Note

The Change command does **not** change the file type. Use Translate to change file type.

Extended directory

The Extended directory command lists the directory of a blocked volume or a set of files in the volume.



Item	Variable Parameter Type	Range Restrictions
file specification	literal	(See the beginning of this chapter)
volume specification	literal	(See the beginning of this chapter)

Semantics

The Extended directory prompt:

```
List_ext what directory ?
```

The Extended directory command requires a legal volume ID or a file specification. Results can be listed to the PRINTER: or to a text file if specified and separated from the first specification by a comma. If no destination is specified the listing defaults to the CONSOLE. Wildcards are available to identify subsets of files on the volume.

The name of the volume is displayed in the upper left-hand corner. To the right, the directory type is displayed. The next two lines contain the date the directory was created, the size of the storage blocks, and whether the listing is in storage order or alphabetical order.

To have directories listed in alphabetical order, include [*] after the directory name. For example:

```
MYDIR:[*]
```

The column entries for each file include: file name, number of blocks used for storage, the file size in bytes, the number of the block where the file starts, the date the file was changed, the type as recognized by the file system, the type-code used by the directory system, the date the file was created and two extension fields.

The last two lines display additional directory information including how many more entries can fit in the directory. The size of a directory is specified when the disc is initialized.

The results can be listed to a printer or a file if you so specify. The destination of the listing is separated from the volume ID or file specification by a comma. If no destination is specified, the listing defaults to the screen. Wildcards are available to specify groups or subsets of files on a mass storage medium.

For example, assuming that MYVOL: is in the disc drive associated with logical unit #3, a listing of all the TEXT files on that volume could be sent to the printer by specifying any of the following in response to the Extended directory prompt:

#3: =TEXT,#6:	specifies volume residing in unit #3; listing to logical unit #6: (PRINTER:)
* =TEXT,PRINTER:	specifies system volume; listing to the PRINTER. Without the colon, the listing would be sent to a DATA file named "PRINTER" on the default volume.
MYVOL: =TEXT,#6	specifies MYVOL: volume; listing to unit #6.

In all cases the "=TEXT" string refers to all files whose names end in TEXT on the specified volume and the listing is sent to the printer.

Listings can also be sent to a file. Use a destination parameter after the the source parameter (separated by a ",") as in the above PRINTER: example. Give a complete file specification. Use the appropriate suffix in the file name. Otherwise, a file of type DATA is produced. For example:

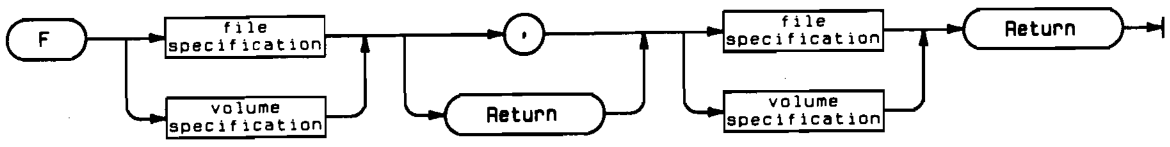
```
List what directory ? #3:,MYVOL:LIST.TEXT
```

or

```
List what directory ? #3:,MYVOL:LIST.ASC
```

Filecopy

The Filecopy command copies a specified file, set of files or volume to the specified destination.



Item	Variable Parameter Type	Range Restrictions
file specification	literal	(See the beginning of this chapter)
volume specification	literal	(See the beginning of this chapter)

Semantics

The Filecopy prompt:

```
Filecopy what file ?
```

The Filecopy command is initiated by pressing (F) and requires two specifications – a source and a destination separated by either a comma (,) or (Return). The source volume must be on-line. The destination volume does not have to be on-line.

Wildcards may be used to specify sets of files. If the equals (=) wildcard is used, the copy is not confirmed before taking place. Also, note that if the equals wildcard is used alone (i.e., without any qualifying strings) then the Filer copies every file on the specified volume. If the question mark wildcard is used, you are asked to verify the transfer of each file meeting the wildcard specification before the Filecopy takes place. Thus, using the wildcard allows you more flexibility and control over the process.

The dollar sign character ($) may be used in the destination specification to indicate that the file(s) will have the same name (or names) as the source file(s). For example, assuming that there are a number of TEXT files on the volume TRIG: and that a second volume named MATH: exists,

```
TRIG:=TEXT,MATH:$
```

This results in all the files on the TRIG: volume whose file names end with the string TEXT being copied to the volume MATH: and given the same name as they have on the TRIG: volume.

Be sure to use either a file name or the $ character when specifying a destination volume. If, in the example above, the destination volume was specified as MATH: instead of MATH:$, the Filer would respond:

```
Destroy directory of MATH: ?
```

If you respond with Y, the directory of that volume gets overwritten. Pressing N aborts the Filecopy command and returns the Filer prompt.

On a system with a single disc drive, the Filecopy command reads the specified file or files into memory, prompts you to remove that volume and insert the destination volume, and then writes the file(s) in memory to the destination volume. Depending on the amount of memory in your computer and the amount of material being copied, you may have to swap discs more than once.

Note

When using the Filecopy command with a single disc drive, wait for the Filer's prompt, then remove the source volume and replace it with the destination volume. Failure to follow this guideline may result in the loss of information from the source volume.

A size specification may be used in the destination description. For example, specifying:

```
MYVOL:FILE,OTHERVOL:FILE[35]
```

would result in the file being written to the first available area on OTHERVOL: that was at least 35 blocks in size.

To make a back-up copy of an entire volume, use the Filecopy command. Simply type in the source volume ID and the destination volume ID. The destination volume must be initialized but does not have to be Zeroed (the directory gets copied from the source volume). The Filer will ask you if you want the directory destroyed. A volume-to-volume copy makes an **exact** copy of the source volume on the destination volume.

Note that having two volumes with the same name on-line at one time is **not** advised. The Filer looks for volumes according to their volume names and may not be able to distinguish one from the other. Thus, the Filer may perform an action on one volume when you wanted the operation to affect the other volume. The Filer warns you whenever this condition exists. If you get a warning, either remove one of the volumes or use the Filer's Change command to change the name of one of the volumes.

You can copy files on one volume to a volume of a different size but you should not use volume IDs alone to do this. If the source volume is larger than the destination volume, the Filer refuses to execute the Filecopy. If the source is smaller than the destination, the destination volume ends up the same size as the source when the operation is through so you lose storage space. It makes an exact duplicate of the source.

The best way to handle copies between different size volumes is to use one of the wildcards. Use the equals wildcard (=) if the destination is larger than the source and the question mark wildcard (?) if the destination is smaller than the source. In the latter case you may have to be selective in your copies since there may not be enough space for all of the files.

When the Filecopy command has finished its task, the screen displays what file(s) or volume has been copied and the Filer prompt appears. The Filecopy command can be aborted before all specifications are complete by pressing (Return) in response to the prompt.

In cases where the destination volume already contains a file with the same name as the file being copied, this prompt is displayed:

```
ANYVOL:XFILE
exists ..., Remove, Overwrite, Neither ? (R/O/N)
```

You have the options:

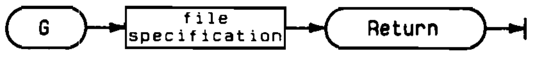
- **Remove** writing the file to the volume and if successful, removing the original.
- **Overwrite** replace the contents of the old file with the new information. The new file has the same starting location as the original.
- **Neither** cancel the operation.

Note

The Filecopy command does **not** change the file type. Use Translate to change file type.

Get

The Get command associates a specified file as the current workfile.



Item	Variable Parameter Type	Range Restrictions
file specification	literal	(See the beginning of this chapter)

Semantics

The Get prompt:

```
Get what file ?
```

The Get command is initiated by pressing (G) and prompts you for a file specification. If a workfile currently exists when the Get command is executed, you are asked if you want to release that file before being allowed to specify a new workfile. Upon receiving the specification, the Filer finds the file (or files) and associates that name with the current workfile. Subsequent operations on the workfile use the specified name. The workfile is generally named `*WORK.TEXT` and/or `*WORK.CODE`.

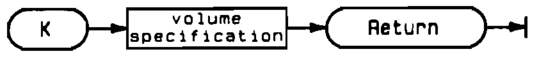
The Get operation assumes that the text version of the specified file has a `.TEXT` suffix. If the text version is ASCII, you must include the `.ASC` suffix. If the text version is DATA, you must include a "`.`" at the end of the file name to prevent the appending of the `.TEXT` suffix.

If both text and code versions of the specified file exist, both are associated with the workfile; if only one exists, the association is made with that file. The Filer reports one of three things: either a text or code file has been loaded, both have been loaded or the file cannot be found on the specified volume.

The Filer is not the only HP TechWriter subsystem where a workfile can be created. The Editor also creates workfiles. Once a workfile exists, it is treated as the default file until it is "released" by the Filer's New command.

Krunch

The Krunch command moves all files on a block structured volume so that all the unused storage space is at the end of the volume.



Item	Variable Parameter Type	Range Restrictions
volume specification	literal	(See the beginning of this chapter)

Semantics

The Krunch prompt:

```
Crunch what directory ?
```

The Krunch command is initiated by pressing `K` and it prompts you for a volume ID. After you respond with a legal volume ID of an on-line block structured volume, it prompts:

```
Crunch directory MKWORK: ? (Y/N)
```

Where `MKWORK:` is whatever volume you specified. Typing `Y` for Yes lets the command continue; `N` for No returns the Filer prompt. The Krunch command executes a sensitive operation – that of moving all the files forward on the disc by reading the files into memory and then writing them back out on the disc in such a manner so as to make all the unused space on the volume contiguous at the end of the disc.

The Krunch command is used when, after repetitive reading and writing to a disc, the available storage space becomes highly fragmented. For example, you can have 100 blocks available on the disc but because they are all in 10 or 15 block chunks, there is not enough contiguous storage space for the system to write a 20 block file to the disc. The Krunch command moves all existing files so that the free space is in a single contiguous block.

If there is the slightest question about the reliability of the medium you are using (because of excessive wear or damage), use the Bad sector command to do a scan of the sector on the volume **before** initiating Krunch. If a bad sector is found, use the Filer's Make command to make a file of type .BAD over the bad sectors. Krunch does not move files of type .BAD. Moving files onto an unreliable area of storage is a good way to lose a file.

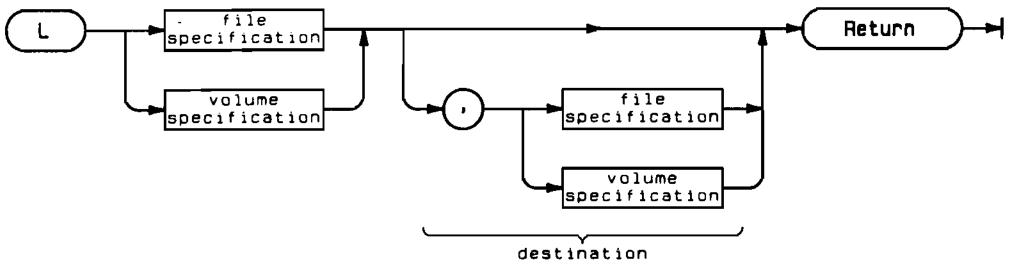
CAUTION

UNDER NO CIRCUMSTANCES SHOULD YOU ATTEMPT TO INTERRUPT THE KRUNCH OPERATION ONCE IT HAS BEGUN. YOU ARE RISKING YOUR DIRECTORY AND THUS, ALL THE INFORMATION CONTAINED ON THAT MEDIUM IF YOU DO SO. DO NOT TOUCH THE POWER SWITCH, THE DOOR ON THE DISC DRIVE OR ATTEMPT TO USE THE KEYBOARD WHILE A KRUNCH IS OCCURRING.

The Krunch command is extremely useful and using it should not worry you. However, because it alters the directory, it is one of the quickest ways to wipe out a volume. The precautions outlined above should help you avoid any problems while using the command.

List directory

The List directory command lists directory information about a block structured volume or one of its subsets.



Item	Variable Parameter Type	Range Restrictions
file specification	literal	(See the beginning of this chapter)
volume specification	literal	(See the beginning of this chapter)

Semantics

The List directory prompt:

```
List what directory ?
```

The List directory command requires a legal volume ID or a file specification. Results can be listed to the PRINTER: or to a text file if specified and separated from the first specification by a comma. If no destination is specified the listing defaults to the CONSOLE. You may use wildcards to specify subsets of files on the volume.

The name of the volume is displayed in the upper left-hand corner. To the right, the directory type is displayed. The next two lines contain the date the directory was created, the size of the storage blocks, and whether the listing is in storage order or alphabetical order.

To have directories listed in alphabetical order, include [*] after the directory name. For example:

```
MYDIR:[*]
```

The column entries for each file include: file name, number of blocks used for storage, the file size in bytes, and the date the file was created or changed.

The last two lines display additional directory information including how many more entries can fit in the directory. The size of a directory is specified when the disc is initialized. You need one 256 byte block for each eight directory entries.

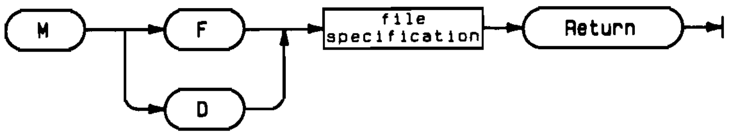
For example, initiating the command by pressing (L), specifying ACCESS: and pressing (Return) results in a display similar to the following:

```
ACCESS:            Directory type= LIF level 1
   created 20-Sep-83 13.57.17 block size=256
    Storage order
   ...file name....    # blks    # bytes  last chng

   FILER                   218      55808 20-Sep-83
   MEDIAINIT               132      33792 20-Sep-83
   FILES shown=5 allocated=5 unallocated=3
   BLOCKS (256 bytes) used=350 unused=223 largest space=223
```

The Extended Directory command gives more information about the files and unused areas on the volume.

Make

The Make command creates files and directories.



Item	Variable Parameter Type	Range Restrictions
file specification	literal	(See the beginning of this section)

Semantics

The Make prompt:

```
Make File or Directory ? (F/D)
```

The Make command is useful primarily in two ways. Files can be made when you need to reserve physical space on a disc, and directories can be made on an SRM system. Please consult the *Pascal 3.0 Workstation System* manual for more information about SRM.

The Make command is **not** required to create files to be used by the various Pascal subsystems. It reserves space only; it in no way initializes or changes the contents of the space. The HP Tech Writer Editor lets you either create or specify any files you need. Users of HP BASIC may quite naturally think that the same function is served by this command as the CREATE command in BASIC (where you must create a file before using it). Thus the distinction between these similar sounding commands is drawn here.

The Make command requires at least a file specification (which includes a volume ID by definition) and accepts an optional size specification. If the (positive integer) size is given, it must follow the file specification on the same line and be enclosed in square brackets. The Filer then creates a file with the specified name and of the specified size on the first area of the volume that has a large enough area of contiguous storage space to meet the size requirements.

When using a size specification to make a file, you must be aware that the size is specified in "number of blocks". The size of all "Make" blocks is 512 bytes – regardless of the directory type. A LIF directory considers a 256 byte sector to be a block. So if you make a file on a LIF volume and specify 500 blocks, it will show up in the directory as 1000 blocks.

For example, assume that there is a volume named MKWORK: on-line that has at least 22 blocks of contiguous and unused space available. Press M to initiate Make, specifying:

```
MKWORK:DUX.TEXT[22]    Enter
```

This results in a file named `DUX.TEXT` being created on the first available area with 22 blocks of the volume MKWORK: and the Filer reporting the following:

```
MKWORK:DUX.TEXT made
```

A subsequent listing of the directory (using the List directory or Extended directory commands) will show a file of the same name with a 44 block size.

The size specification may be omitted in which case the Filer creates the specified file using the largest unused area on the disc (i.e., the largest contiguous storage space on the disc will be allocated to the file). We recommend that you specify the size you want the file to be.

There are two special cases of size specification worth knowing about. The first is the number zero enclosed in brackets `[0]` which is the same as omitting the size specification altogether – the Filer uses the largest space available. The second case is the asterisk character enclosed in brackets `[*]` which tells the Filer to make the file's size either the second largest area on the disc or half of the largest area, whichever is greater.

The Make command is useful if you must rebuild a file that was lost on a disc.

1. You must know its size and where it was located.
2. Then make TEMP files (TEMP1, TEMP2 etc.) over all the unused spaces on the disc that are as large or larger than the file you'll be making.
3. Then make a file of the proper type over the lost file to recover it.
4. Finally, use the Filer's Remove command to remove all the TEMP files.

An Extended directory listing can help you determine the location and size of unused areas on the disc.

New

The New command releases or clears the workfiles.

Semantics

The New command requires no specifications. Upon pressing N to initiate the command, it clears the workfile unless the workfile has been updated since the last Save command. If this is the case, the prompt appears:

```
Throw away current workfile? (Y/N)
```

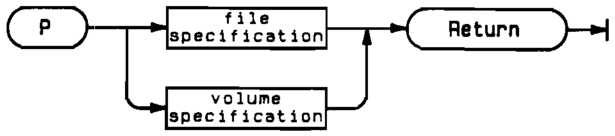
Responding by pressing N for No allows you to use the Save command to write the file to a volume; Y for Yes clears the current workfile area. When the Filer executes the New command, it will respond with:

```
Workfile cleared
```

You can check the status of the workfile before using New with the What command. The What command gives you the name and status (saved or not) of the current workfile.

Prefix

The Prefix command changes the default volume to the one specified.



Item	Variable Parameter Type	Range Restrictions
volume specification	literal	(See the beginning of this chapter)

Semantics

The Prefix prompt:

```
Prefix to what directory ?
```

The Prefix command is initiated by pressing (P) and requires a volume ID. The command allows you to specify a new default volume – the one where the Filer searches for file specifications when a volume name is not specified. The volume must be block structured (one used for mass storage) but does not have to be on-line. The current prefix (i.e., default) volume can be obtained by responding to the Prefix prompt with a colon (:). (The Volumes command may also be used).

When the command executes, the screen displays the message:

```
Prefix is MKWORK:
```

Where MKWORK: is the name of the current default prefix. The Prefix command saves keystrokes if you are doing a lot of file accessing on a particular volume.

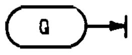
Filer commands which request a volume ID may be answered with the colon character (:) which specifies the current default volume.

It is possible to set the default prefix to a flexible disc drive, regardless of the volume inside. This is done by typing the following while the drive door is open.

```
#3: (Return)
```

Quit

The Quit command exits the Filer subsystem and returns control to the HP TechWriter Environment.

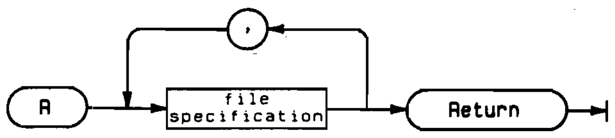


Semantics

The Quit command has no parameters and no specifications of any type are needed. Pressing Q exits the Filer and displays the HP Techwriter command prompt.

Remove

The Remove command purges specified files from a directory.



Item	Variable Parameter Type	Range Restrictions
file specification	literal	(See the beginning of this chapter)

Semantics

The Remove prompt:

```
Remove what file ?
```

The Remove command is initiated by pressing R and requires a file specification. The command removes the specified file from the directory, updates the directory and reports the action it has performed. Wildcards may be used to specify several files to be removed. If the equals wildcard (=) is used in the file specification, the Filer reports the specified file or files and then prompts:

```
Proceed with remove ? (Y/N)
```

This is the last chance you have to change your mind about the removal. Pressing N aborts the operation and no files are removed. Pressing Y removes those files meeting the wildcard specification from the directory. The process is not always reversible. However, the Make command can sometimes be used to recover a removed file.

Note

The Filer considers the file specification = to specify ALL the files on the default volume and MKWORK: = to specify ALL the files on the MKWORK: volume. If you use the wildcard in this form and respond to the Filer's prompt (`Proceed with remove ? (Y/N)`) with a Y for Yes, every file on the directory of the specified volume is removed. Responding with a N for No aborts the operation. Wildcards can be hazardous to your files – watch the prompts.

Specifying a single file (of an on-line volume of course) in response to the Remove prompt results in the removal of that file from the directory and a report that the file has been removed. Once the (Return) key is pressed following the file specification (unless wildcards are used), that file is gone.

While the use of the equals wildcard (=) results in being prompted for whether or not you want the directory updated, the question mark wildcard (?) acts slightly differently. It allows you to be more selective in your removal. Given the volume PROCESS: containing the files:

```
NOVMEMO.TEXT
MARKLTR.TEXT
PARSER.TEXT
PARSER.CODE
GARBAGE.TEXT
```

The specification PROCESS:?TEXT in response to the Remove prompt results in the screen clearing and the following message appearing.

```
Remove NOVMEMO.TEXT ? (Y/N)
```

Answering with either a (Y) or (N) results in the next prompt appearing below the first:

```
Remove MARKLTR.TEXT ? (Y/N)
```

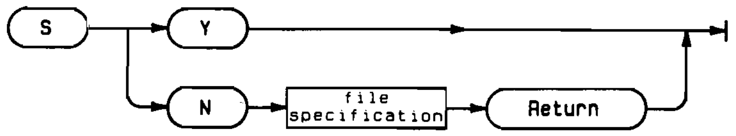
The process continues until you have been prompted for all the TEXT files on the PROCESS: volume and then the final prompt appears:

```
Proceed with remove ? (Y/N)
```

You may be respond with either a (Y) (for Yes) or (N) (for No). The files are not actually removed until this prompt is answered with a (Y). The ? wildcard thus allows you to be both selective and relatively safe about your file removals.

Save

The Save command saves the current workfile on the specified volume.



Item	Variable Parameter Type	Range Restrictions
file specification	literal	(See the beginning of this chapter)

Semantics

The Save command is initiated by pressing `S` and may or may not require a file specification. If the workfile was never updated, it is automatically saved with the original name.

If the workfile was previously named using the Save command, or originally obtained using the Get command, then the Filer prompts:

```
Save as PREVIOUS.TEXT ? (Y/N)
```

Where `PREVIOUS.TEXT` is the name previously associated with the workfile. Responding with a `Y` for Yes results in either a CODE or a TEXT file or both, (depending on what is in the workfile) of that name being removed and replaced with the current workfile.

If the workfile is not named, or if you answer `N`, the Filer prompts:

```
Save as what file ?
```

When naming the file, the following conventions apply to the type of the file:

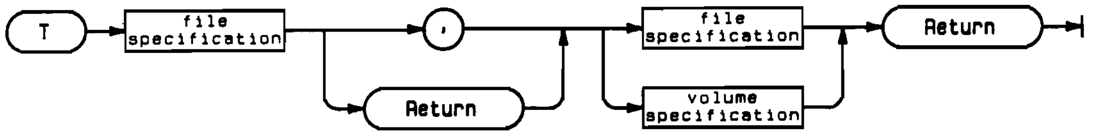
1. If a standard suffix is recognized, the workfile is either Filecopied, Translated or Changed (on the system volume) to the file name and type.
2. If no suffix is recognized, the file type is `.TEXT` by default.
3. If no suffix, but a "." is found, the file type is DATA.
4. The `.CODE` file is created by removing the suffix (if there is one) and adding `.CODE` to the file name.

The Filer displays that the file is now saved.

To find out what the current name and state (saved or not) of the workfile is, use the What command.

Translate

The Translate command converts text files between the TEXT, ASCII, and DATA formats.



Item	Variable Parameter Type	Range Restrictions
file specification	literal	(See the beginning of this chapter)
volume specification	literal	(See the beginning of this chapter)

Semantics

The Translate prompt:

```
Translate what file ?
```

The Translate command is initiated by pressing (T) and requires two specifications – a source and a destination separated by either a comma (,) or a carriage return (press (Return)). The source specification can be any block structured volume, any file or any group of files on a volume. The destination specified can be any of the above and may also be a non-block structured volume (i.e., the PRINTER: or CONSOLE:). Non-block structured volumes (like the PRINTER:) are assumed to be on-line.

Wildcards may be used to specify sets of files but if a wildcard is used in the source specification, either a wildcard or the $ character (discussed below) must be included for the destination. If the equals wildcard (=) is used, the translate is not confirmed before taking place. Also, note that if the = wildcard is used alone (i.e., without any qualifying strings such as LIBR, TEXT, etc.) then the Filer Translates every file on the specified volume. If the question mark wildcard (?) is used, you are asked to verify the translate of each file meeting the wildcard specification before the Translate takes place. Thus, using the ? wildcard allows you more flexibility and control over the process.

The dollar sign character ($) may be used in the destination specification to indicate that the file(s) will have the same name(s) as the source file(s). For example, assuming that there are a number of TEXT files on the volume `TRIG:` and that a second volume named `MATH:` exists,

```
TRIG:=TEXT,MATH:$
```

results in all the files on the `TRIG:` volume whose file names end with the string `TEXT` being translated to the volume `MATH:` and given the same name as they have on the `TRIG:` volume.

On a system with a single disc drive, the Translate command reads the specified file or files into memory, prompts you to remove that volume and insert the destination volume, and then writes the file(s) in memory to the destination volume. Depending on the amount of memory in your computer and the amount of material being translated, you may have to swap discs more than once.

Note

When using the Translate command with a single disc drive, wait for the Filer's prompt before removing the source volume and replacing it with the destination volume. Failure to follow this guideline may result in the loss of information from the source volume.

The Translate command allows the translating of files or groups of files to non-block structured devices like the `PRINTER:` and `CONSOLE:`. Only ASCII files should be sent to printers since other files are not generally human readable.

When the Translate command has finished its task, the screen displays what file(s) has been translated and the Filer prompt appears. The Translate command can be aborted before all specifications are given by pressing (Return).

In cases where the destination volume already contains a file with the same name as the file being Translated, this prompt is displayed:

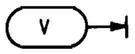
```
ANYVOL:XFILE
exists ..., Remove, Overwrite, Neither ? (R/O/N)
```

You have the options:

- **Remove** writing the file to the volume and if successful, removing the original.
- **Overwrite** replace the contents of the old file with the new information. The new file will have the same starting location as the old file.
- **Neither** cancel the operation.

Volumes

The Volumes command lists the volumes currently on-line.



Semantics

The Volumes command requires no specifications. When V is pressed, the Filer displays information about all on-line volumes.

This is a typical display generated by the Volumes command:

```
Volumes on-line:
  1   CONSOLE:
  2   SYSTERM:
  3 * ACCESS:
  4 * MINI4:
  6   PRINTER:
Prefix is - ACCESS:
```

The number on the far left is the logical unit number associated with the volume. The * character in the second column indicates the system volume which is always block structured. The * character indicates other block structured volumes currently on-line. The remaining volumes (shown with no character in the second column) are non-block structured. The last line of the display shows the current default volume, where the system looks for a file when no volume has been specified.

The above configuration shows two 3.5 or 5.25-inch flexible disc drives associated with units #3 and #4.

What

The What command displays the name and state (saved or not) of the workfile.

Semantics

The What command is initiated by pressing W and requires no other input. The command shows the name of the current workfile or, if no workfile exists, the Filer responds with:

```
No workfile
```

The Filer also shows whether or not the workfile has been saved since the last update to the file.

For Example, suppose you had two files named `INFRARED.TEXT` and `INFRARED.CODE` on the default or prefix volume. Assume that you used the Filer's Get command and specified `INFRARED` to associate the files with the workfile. If you then edited the TEXT version of that file (using the Pascal Editor), returned to the Filer and executed the What command, the screen would display:

```
Workfile is INFRARED (not saved)
```

because the workfile was changed since the last time a Save command was executed.

Saving the workfile does not change the fact that the workfile exists. It is still there. The New command must be used to clear the workfile. Saving the workfile is not remembered between separate sessions of the Filer. If you Save the workfile during the current Filer session, a New command immediately clears the workfile. If you Save it, quit the Filer and then return to use the New command, the Filer will ask:

```
Throw away current workfile ? (Y/N)
```

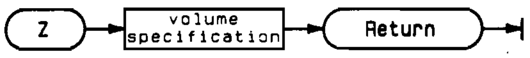
even though you saved it during the previous Filer session and haven't updated it since.

Note

Because the file system works with volume names, a LIF volume whose name is all blanks (ASCII spaces) will not be recognized as a valid volume.

Zero

The Zero command creates an empty directory on the specified volume.



Item	Variable Parameter Type	Range Restrictions
volume specification	literal	(See the beginning of this chapter)

Semantics

The Zero prompt:

```
Zero directory (NOT valid on SRM type units)
Zero what directory ?
```

The Zero command is initiated by pressing Z and requires the volume ID of a block structured volume. The volume must be formatted using the Pascal utility program `MEDIAINIT.CODE` supplied on the `ACCESS:` volume.

Since the Zero command creates new empty directory on the volume, you will be prompted:

```
Destroy THISVOL: ? (Y/N)
```

Responding with a N for No aborts the command and returns the Filer prompt.

If you answer Y, the next prompt is:

```
Number of directory entries (8) ?
```

The number in the parentheses is the number in the existing directory. Respond with Return if that is the number you want. If there is no number in parentheses, Return causes the default number for that directory type (80 for LIF) to be put on the disc.

The next prompt is:

```
Number of bytes (270336) ?
```

It is asking for the logical size of the disc (the extent to be managed by the directory). The number in the parenthesis is the number in the existing directory or the default for that disc. Press Return to use the displayed number.

The next prompt is:

```
New volume name ?
```

The Filer is asking for a legal volume name. Volume name formats vary with different directory structures. LIF directories allow up to six upper or lower case characters.

An answer of (Return) aborts the Zero command.

After typing a volume name, the final prompt appears:

```
NEWSTUF: correct ?
```

Responding with (N) aborts the Zero command. Responding with (Y) results in the message:

```
NEWSTUF: zeroed
```

Where `NEWSTUF:` is the name of the new volume. The Filer prompt reappears when the operation is complete.

Differences Between HP TechWriter and the Pascal Text Editors

Appendix A

If you own a Pascal operating system and have looked at the HP TechWriter Editor, you have probably observed that the HP TechWriter Editor is in many ways similar to the text Editor that came with your system. The HP TechWriter Editor has many enhancements and additional capabilities that are not available in the Pascal operating system Editor. These are summarized below. They are divided into three sections. The first addresses the changes and additions to the keystroke commands. The second section summarizes the document commands available from HP TechWriter. Document commands do not exist in the Pascal operating system Editor at all. The third section summarizes the other differences between the two Editors.

Keystroke Commands

Adjust	None.
Copy	None.
Delete	None.
Find	When Finding with a finite repeat factor, if the Editor finds at least one occurrence of the specified string, but does not find the number requested, it tells you how many it found. The environment parameter `Ignore case` works for Literal finds.
Help	New. This did not exist at all in the Pascal Editor.
Insert	While in Document mode, Insert margins the whole paragraph, rather than just the remainder from the cursor position on. If you Insert several paragraphs (delimited by either blank lines or command lines) the Editor margins all of them, not just the last one inserted. Local margin specifications are honored. Right arrows create spaces when used during Inserts.
Jump	Marker names may only have a maximum of eight characters.
List	New. This did not exist at all in the Pascal Editor.
Margin	Right Justification can be enabled by setting the environment parameter `Justify` to `True`. This causes spaces to be inserted into the lines such that the right margin is smooth. A Margin command can be preceded by a repeat factor to margin that many paragraphs. The direction the paragraphs extend depends on the current cursor direction. Local margin specifications are honored.

Local margin specifications can be specified by the document command `^^M`. These local specifications can set left-, right-, and paragraph margins, right justification, or disable margining completely.

Locking margins (local margin specifications which remain in effect until explicitly turned off) can be specified by the secondary keyword `ON`.

Page — None.

Quit — Another option has been added: Archive. When you archive a file, an "`@`" is appended to the name of the file which already exists on the mass storage device, and the current file is stored in a file whose name is the unchanged name.

Replace — When Replacing with a finite repeat factor, if the Editor replaces at least one occurrence of the specified string, but does not find the number requested, it tells you how many it replaced.

The environment parameter `Ignore case` works for Literal replaces.

When doing a Replace with Verify, the HP TechWriter Editor requires that you press R, the space bar, or Shift-Select. In the Pascal Editor, everything was considered a blank (no replace) except R and Shift-Select.

Set — The environment has several new options:

- `Justify` This allows right justification (see Margin).
- `Underline ch` This defines the character which causes underlining to be turned on or off.
- `Escape ch` This defines the character which, when followed by three digits, sends that ASCII character to the printer. This allows you to specify escape sequences to access capabilities specific to your printer.
- `Draw figures` When `True`, this tells the Editor to access drawing files referenced in the file, and draw the pictures. If `False`, nothing is drawn.

On the bottom line of the environment display, the type of the file is displayed: `TEXT`, `ASC`, `DATA`, or `new`.

When specifying a parameter in Set, Shift-Select aborts the entry and reinstates the previous value.

Verify — Redraws any figures referenced on the screen (if environment parameter `Draw figures` is `True`).

eXchange — None

Zap — Always requires verification.

Document Commands

A Document Command is a command which is placed into the text file itself, rather than a command executed by pressing a key while using the Editor. The first two characters in a document command are two occurrences of the command character, as defined in the environment. The third character in the document command must be the character which indicates the command desired. These characters must be contiguous, and they must be the first three non-blank characters in a line, although they may be preceded by one or more blanks.

^^C	Specifies items to be put into the table of contents or defines the table of contents title. The items may or may not also remain in the printed document. Tables of contents are generated by the Lister.
^^D	Allows drawings to be plotted on the screen and printed. You can specify the file containing the image, local margins (drawing limits), isotropic or anisotropic images, and horizontal and vertical justification within the drawing limits. A box may be drawn around the image.
^^J	Allows you to specify the number of lines to skip (jump) on the output, and to place lines of text at the bottom of the page.
^^L	Enables you to turn a listing on and off.
^^M	Allows you to specify local margins which are different from those set in the global environment. Local specifications affect only the next one paragraph unless the secondary keyword `ON` is included. In a `^^M` command, you can specify left margin, right margin, paragraph margin, and justification true/false.
^^P	With this command, you can specify an absolute page number of the next page of output (e.g., "the page number is 12"), specify a relative page number of the next page of output (e.g., "add 6 to the current page number"), and cause a conditional or unconditional page feed.
^^S	Allows you to specify line-spacing such as: single spaced, double spaced, triple spaced, etc., or zero spaced (for overprinting).
^^T	Specifies the contents of the page footer. The page footer is a line of text which is printed at the bottom of every page of output.

Other Changes

- When in Program mode, the computer beeps when the cursor enters the column eight columns to the left of the *right margin*, rather than eight columns to the left of the right edge of the screen.
- The computer beeps for more exceptional conditions, to prevent you from typing a long time before discovering it is waiting for you to clear a condition before continuing. Some of these places are: trying to eXchange past the end of a line, trying to back up farther than you went forward in Insert, and giving a illegal delimiter in a Find or Replace.
- In any area where the Editor is waiting for input, a Shift-Select will abort the entry immediately, rather than having to press Clear line, Return.

A Look Inside the Editor

Appendix B

This section contains some details about how the Editor works. It includes information on memory and file sizes and on file types and file naming conventions. At the end of the section there are two procedures for working with blocks of text that are too large to fit in the copy buffer.

Memory and File Sizes

When the Editor is entered, the current text file is stored in the computer's read/write memory. All changes that occur to a text file (including text creation) take place in this memory and only become permanent when the Editor is exited and the contents of the text file are written from memory to a mass storage medium such as a flexible disc.

The maximum size of the text files that can be accessed or created by the Editor depends on the memory configuration of your system. This size can easily be determined using the Set Environment command. The two environment headings, "bytes used" and "available", should be added together. The sum equals the maximum size (in bytes) of the text files which can be handled by the Editor.

If your text file approaches the maximum size while you are doing an insertion, the Editor displays the following message:

```
>ERROR: Finish the insertion  <space> continues.
```

This tells you that you are nearing the Editor's memory limits. If, after finishing the insertion, you attempt to initiate the Insert command again, the Editor informs you:

```
>ERROR:  No room to insert.  <space> continues.
```

There is a procedure included later in this appendix which you can use to help you work around the Editor memory limits (whatever they may be on your machine). The section is called *Splitting Big Files.*

File Names and Types

The Editor can read and write three types of files. The predominant file type is TEXT. The others are DATA and ASCII. TEXT files contain ASCII characters and are structured in a particular way.

In every text file, the first two blocks (or 1024 bytes) are reserved for information about the environment settings, list environment, the locations of markers in the file and other information the Editor needs to work with that file. Since the Editor allocates mass storage in two-block increments, text files always contain an even number of blocks. Also, because the Editor reserves the first two blocks for file information, a file with only a single character will take up four blocks (or 2048 bytes) of storage space on a mass medium.

It is possible to create a file that does not have `.TEXT` appended to the end of the file name. If, when exiting the Editor and specifying a file name using the Write option, you place a period at the end of the file name, the Editor does not append `.TEXT` to the file name. The text, but not the environment information, is stored on a file of type DATA.

If you want to access this file with the Editor, you must specify the file name followed by a period when entering the Editor. If you do not use the trailing period, the Editor appends `.TEXT` to the name you type in and looks for a file with that name on the mass storage medium.

For example, suppose, when exiting the Editor, you answer the prompt for a file name with `MyFile.`. Notice the period following the name. The Editor saves the file with the name `MyFile` (it strips off the period) and does not append the `.TEXT` suffix. If you enter the Editor and want that file, you must specify `MyFile.` . If you instead specify `MyFile` (i.e., leave off the period), the Editor appends `.TEXT` to the name you typed and looks for a file with the name `MyFile.TEXT`. It may even find a file with that name, but it will be a different file than the one saved by specifying `MyFile.`.

ASCII files are structured differently. ASCII files on LIF discs are compatible with the BASIC and HPL language systems that run on your computer. ASCII files are created by writing to a file whose name ends in the suffix `.ASC`. When writing ASCII files, the Editor's environment information is not stored along with the text.

The HP TechWriter system as a whole can create several types of files:

- The HP TechWriter text Editor can create `.TEXT`, `.ASC`, and `Data` files with the Quit command. These are for storage of text. Output spool files, which can be created by the List command are nothing more than ASCII-type files which are ultimately destined for a printer.
- The Picture Processor takes files of Hewlett-Packard Graphics Language (HPGL) commands and translates them into a file which the HP TechWriter Editor and Lister can read.
- The Lister creates, at your request, a list file; that is, a file containing the names of the files to print as a document and other pertinent parameters. The Lister automatically appends "`.LS`" to the file names unless you place a period at the end of the file name. A list file is shown in an extended directory listing as type "`Data`", regardless of whether it has `.LS` in its name.

Moving Very Large Sections of Text

If you want to move a section of text too large to fit in the copy buffer from one place in your file to another, here is a way to do it.

1. Set a marker (see the Set command in the Editor Reference chapter), say `A`, at the beginning point of the section of text you wish to move.
2. Set another marker , say `B`, at the end point of the section of text you wish to move.
3. Re-store the file as a `.TEXT` file on a disc and return to the Editor. *The file must be a* `.TEXT` *file; if it is not, the marker locations are lost.*
4. Delete the text between the two markers. If you get the message:

```
No room to copy deletion. Delete anyway? (y/n)
```

   go ahead and say yes.

5. Now move the cursor to the position to which you want to move the text, and copy your file (the one you just saved) from marker `A` through marker `B` into the Editor at the cursor position. Remember, the text was deleted from the *memory* copy, not from the mass storage copy.

Remember, only `.TEXT` files store the Editor's environment along with the file. Marker locations are not stored for `.ASC` or `Data` files, and this technique will not work for those file types.

Splitting Big Files

A good way to split large files into two pieces is to use the Zap command. A typical sequence to do this follows:

1. Go to the breaking point of the large file, and set a marker. Let's call it "`Z`".
2. Re-store the file (as a `.TEXT` file, so the marker definition is not lost).
3. Return to the Editor, jump to marker "`Z`", and press A Select to set the anchor point at the same point as marker "`Z`".
4. Jump to the end of the file (J E) and Zap the last half (we want to keep the first half) of your file (Z Y: Zap? Yes.) If you get the message, `No room to copy deletion. Delete anyway? (y/n)`, press Y.
5. Store this file under a *new name*, then exit and reenter the Editor with the file you saved in Step 2.
6. Jump to marker "`Z`", and Zap the first half (we want to keep the last half this time) of your file (Z Y: Zap? Yes.) We don't need to set the anchor this time because it is at the first character in the file until set elsewhere.
7. Store this file under another new name, and you're done.

Making Your Own Pictures

Appendix C

If you want to create your own pictures to illustrate your documents, you can use any of four methods:

- Produce pictures with a graphics editor. You can get a list of the graphics editors which work with HP TechWriter from your local HP Sales Representative and purchase the one which best suits your needs.
- Plot to file from programs written in Pascal version 2.1 or later.
- Plot to file from programs written in BASIC 3.0.
- For simple pictures, use your HP TechWriter Editor to create a plot file. The plot file contents are described in detail in the *Picture Processor Reference* of this manual and an example plot file is shown later in this section.

The plot files generated by these methods must be translated through the HP TechWriter Picture Processor. A step-by-step example of how to use the Picture Processor is included in the *Getting Started* chapter, and there is more information on the Picture Processor in the *Picture Processor Reference* chapter of this manual.

The following sections discuss the possible sources for plot files in a little more detail.

Graphics Editors

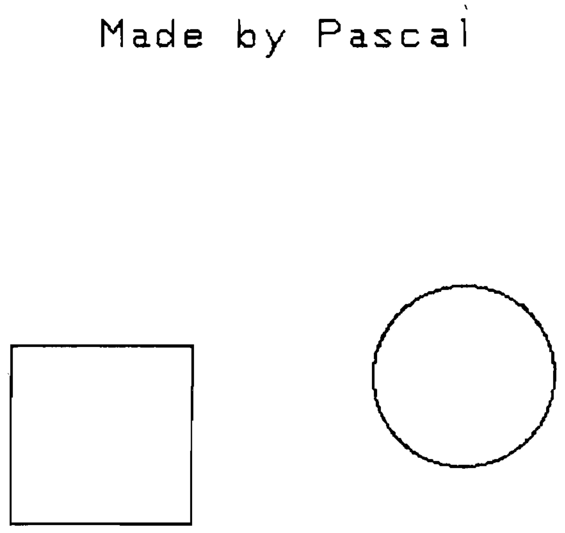
Contact your Hewlett-Packard Sales Representative for a list of the available graphics editors that work with HP TechWriter. The documentation for each of these graphics editors will describe how to send plotting commands to files.

Pascal Programs to Produce Pictures

If you have a Pascal operating system version 2.1 or later, you can write programs to draw pictures in Pascal. For example, here is a listing of a Pascal program which draws a square and a circle, and prints the text "Made by Pascal".

```
program Pascal(output);
import dgl_lib;
var
  Error:                integer;                { Error return variable }
  Degrees:              integer;
  Radians:              real;
begin
graphics_init;
if true  then display_init(3,0,Error)           {Write to the screen or}
else display_finit('Pascal','9872',0,Error);    {plot to a file}
if Error=0 then begin
  set_window(0,100,0,100);
  move(0,0);                                    { \    Draw a square    }
  line(30,0);                                   {  \   with side=30,    }
  line(30,30);                                  {   >  in lower left    }
  line(0,30);                                   {  /   corner of        }
  line(0,0);                                    { /    screen.          }
  for Degrees:=0 to 360 do begin                        { \   Draw a    }
    Radians:=Degrees*3.1415926535897/180;               {  \  circle,   }
    if Degrees=0 then move(90,25)                       {   > radius 15,}
    else line(75+15*cos(Radians),25+15*sin(Radians));   {  /  center at }
  end;  {for degrees}                                   { /   75, 25.   }
  set_char_size(4.5,8);                         { \   Write the label   }
  move(15,80);                                  {  >  identifying the   }
  gtext('Made by Pascal');                      { /   source language.  }
  display_term;                     {  Terminate display or close file. }
end  {error=0?}
else writeln('Error: ',Error:0);
graphics_term;                          {  Terminate graphics package.  }
end.
```

Here is the output of the above program:

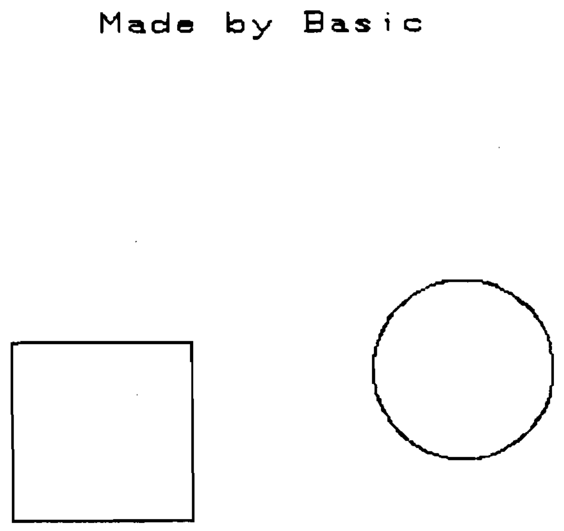


BASIC 3.0 Programs

If you have a BASIC 3.0 Language system, you can write programs to draw pictures in BASIC. For example, here is a listing of a BASIC program which draws a square and a circle, and prints the text "Made by BASIC". It is deliberately designed to do the same operations as the Pascal program just covered, so you can compare the statements.

```
10    ! Program "BasicProg".
20    INTEGER Degrees
30    GINIT
40    IF 1 THEN
50      PLOTTER IS CRT,"INTERNAL"          ! Write to the screen
60      GRAPHICS ON                        !    or
70    ELSE
80      CREATE BDAT "Basic",10             ! write to a file
90      PLOTTER IS "Basic","HPGL"
100   END IF
110   SHOW 0,100,0,100
120   MOVE 0,0                                   ! \    Draw a square
130   DRAW 0,30                                  !  \   with side=30,
140   DRAW 30,30                                 !   >  in lower left
150   DRAW 30,0                                  !  /   corner of
160   DRAW 0,0                                   ! /    screen.
170   DEG                                ! Use degrees instead of radians
180   FOR Degrees=0 TO 360                              ! \
190     IF Degrees=0 THEN                               !  \   Draw a
200       MOVE 90,25                                    !   \  circle,
210     ELSE                                            !    > radius 15,
220       DRAW 75+15*COS(Degrees),25+15*SIN(Degrees)    !   /  center at
230     END IF                                          !  /   75, 25.
240   NEXT Degrees                                      ! /
250   CSIZE 6,.7                          ! \   Write the label
260   MOVE 15,80                          !  >  identifying the
270   LABEL "Made by Basic"               ! /   source language.
280   GINIT                     ! Tie up loose ends or close file.
290   END
```

Here is the output of the above program:



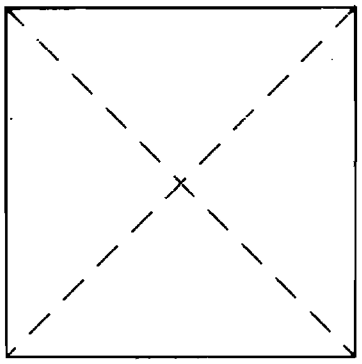
Making a Plot File Using the Editor

The practice of using the Editor directly to make a plot file involves putting the same commands into a file that graphics statements from a program would. For very simple pictures such as the one shown below you can generate pictures in this way.

Plot files are written in *Hewlett-Packard Graphics Language* (HPGL). The Picture Processor understands a short subset of the HPGL command set. The example below uses all of the HPGL commands that the Picture Processor understands. This example draws a box with an "X" in the middle. The comments to the right of the commands should *not* be part of the file; they are merely annotation.

```
PU              (Pick pen up)
PA10,10         (Plot absolute to 10,10: move to upper right corner)
PD              (Put pen down)
PA10,0          (Plot absolute to 10,0: draw to lower right corner)
PA0,0           (Plot absolute to 0,0: draw to lower left corner)
PA0,10          (Plot absolute to 0,10: draw to upper left corner)
PA10,10         (Plot absolute to 10,10: draw to upper right corner)
LT3             (Select line type 3)
PA0,0           (Plot absolute to 0,0: draw to lower left corner)
PU              (Pick pen up)
PA10,0          (Plot absolute to 10,0: move to lower right corner)
PD              (Put pen down)
PA0,10          (Plot absolute to 0,10: draw to upper left corner)
```

Here is the output of the above commands:



Printer Characteristics

Appendix D

HP TechWriter supports a number of printers. The HP TechWriter Editor and Lister first draw pictures on the screen according to sizes defined in lines and text columns by the drawing commands, then print the pictures. Because different printers have different print densities, the size of the printout of pictures will be different from printer to printer, and it may appear to shrink or stretch with respect to the text you saw on the screen. Also, some printers can not print text and graphics side-by-side.

The table below shows the list of the printers HP TechWriter offers, and a comparison between them. The "1:1 Match" column lists the computer or computers with the best vertical picture density match for this printer. The "Maximum picture width" column lists, in inches, the largest width picture that can be produced on this printer at this time. The "Side-by-side" column indicates whether or not this printer can print text and graphics side by side.

Printer	1:1 Match	Maximum picture width	Side-by-side
2631G	9816,9826	6 5/8	yes
2671G	9836	5 3/8	no
2673A	9836	5 3/8	yes
9876A	9816,9826	6 1/4	yes
ThinkJet	9836	5	yes
2932A	9836	5 3/8	yes
2933A	9836	5 3/8	yes
2934A	9836	5 3/8	yes
82906A	9816,9826	6 5/8	yes

LIF Directory Structure

Appendix
E

On a LIF (Logical Interchange Format) directory, the Pascal system codes all the file names that end in a standard suffix. The coding scheme removes the period, appends the first character of the suffix to the file name, and pads the file name to ten characters with "_" (underscore). This allows you to specify file names up to 15 characters. They are encoded to the maximum ten characters for the LIF directory.

In the case of the file name `File.TEXT`, , the first character of the suffix is the fifth character so five "_" characters were added. This coding mechanism is invisible as long as you are using the Pascal system. When you catalogue your disc with other language systems, the coded version of the file name is observed. In our case, `File.TEXT` would be encoded `FileT_____` for storage onto a LIF directory.

Appendix F

US ASCII Character Codes (0-63)

ASCII	EQUIVALENT FORMS				HP-IB
Char.	Dec	Binary	Oct	Hex	
NUL	0	00000000	000	00	
SOH	1	00000001	001	01	GTL
STX	2	00000010	002	02	
ETX	3	00000011	003	03	
EOT	4	00000100	004	04	SDC
ENQ	5	00000101	005	05	PPC
ACK	6	00000110	006	06	
BEL	7	00000111	007	07	
BS	8	00001000	010	08	GET
HT	9	00001001	011	09	TCT
LF	10	00001010	012	0A	
VT	11	00001011	013	0B	
FF	12	00001100	014	0C	
CR	13	00001101	015	0D	
SO	14	00001110	016	0E	
SI	15	00001111	017	0F	
DLE	16	00010000	020	10	
DC1	17	00010001	021	11	LLO
DC2	18	00010010	022	12	
DC3	19	00010011	023	13	
DC4	20	00010100	024	14	DCL
NAK	21	00010101	025	15	PPU
SYNC	22	00010110	026	16	
ETB	23	00010111	027	17	
CAN	24	00011000	030	18	SPE
EM	25	00011001	031	19	SPD
SUB	26	00011010	032	1A	
ESC	27	00011011	033	1B	
FS	28	00011100	034	1C	
GS	29	00011101	035	1D	
RS	30	00011110	036	1E	
US	31	00011111	037	1F	

ASCII	EQUIVALENT FORMS				HP-IB
Char.	Dec	Binary	Oct	Hex	
space	32	00100000	040	20	LA0
!	33	00100001	041	21	LA1
"	34	00100010	042	22	LA2
#	35	00100011	043	23	LA3
$	36	00100100	044	24	LA4
%	37	00100101	045	25	LA5
&	38	00100110	046	26	LA6
'	39	00100111	047	27	LA7
(	40	00101000	050	28	LA8
)	41	00101001	051	29	LA9
*	42	00101010	052	2A	LA10
+	43	00101011	053	2B	LA11
,	44	00101100	054	2C	LA12
−	45	00101101	055	2D	LA13
.	46	00101110	056	2E	LA14
/	47	00101111	057	2F	LA15
0	46	00110000	060	30	LA16
1	49	00110001	061	31	LA17
2	50	00110010	062	32	LA18
3	51	00110011	063	33	LA19
4	52	00110100	064	34	LA20
5	53	00110101	065	35	LA21
6	54	00110110	066	36	LA22
7	55	00110111	067	37	LA23
8	56	00111000	070	38	LA24
9	57	00111001	071	39	LA25
:	58	00111010	072	3A	LA26
;	59	00111011	073	3B	LA27
<	60	00111100	074	3C	LA26
=	61	00111101	075	3D	LA29
>	62	00111110	076	3E	LA30
?	63	00111111	077	3F	UNL

US ASCII Character Codes (64-127)

ASCII Char.	EQUIVALENT FORMS				HP-IB
	Dec	Binary	Oct	Hex	
@	64	01000000	100	40	TA0
A	65	01000001	101	41	TA1
B	66	01000010	102	42	TA2
C	67	01000011	103	43	TA3
D	68	01000100	104	44	TA4
E	69	01000101	105	45	TA5
F	70	01000110	106	46	TA6
G	71	01000111	107	47	TA7
H	72	01001000	110	48	TA8
I	73	01001001	111	49	TA9
J	74	01001010	112	4A	TA10
K	75	01001011	113	4B	TA11
L	76	01001100	114	4C	TA12
M	77	01001101	115	4D	TA13
N	78	01001110	116	4E	TA14
O	79	01001111	117	4F	TA15
P	80	01010000	120	50	TA16
Q	81	01010001	121	51	TA17
R	82	01010010	122	52	TA18
S	83	01010011	123	53	TA19
T	84	01010100	124	54	TA20
U	85	01010101	125	55	TA21
V	86	01010110	126	56	TA22
W	87	01010111	127	57	TA23
X	88	01011000	130	58	TA24
Y	89	01011001	131	59	TA25
Z	90	01011010	132	5A	TA26
[	91	01011011	133	5B	TA27
\	92	01011100	134	5C	TA28
]	93	01011101	135	5D	TA29
^	94	01011110	136	5E	TA30
_	95	01011111	137	5F	UNT
`	96	01100000	140	60	SC0
a	97	01100001	141	61	SC1
b	98	01100010	142	62	SC2
c	99	01100011	143	63	SC3
d	100	01100100	144	64	SC4
e	101	01100101	145	65	SC5
f	102	01100110	146	66	SC6
g	103	01100111	147	67	SC7
h	104	01101000	150	68	SC8
i	105	01101001	151	69	SC9
j	106	01101010	152	6A	SC10
k	107	01101011	153	6B	SC11
l	108	01101100	154	6C	SC12
m	109	01101101	155	6D	SC13
n	110	01101110	156	6E	SC14
o	111	01101111	157	6F	SC15
p	112	01110000	160	70	SC16
q	113	01110001	161	71	SC17
r	114	01110010	162	72	SC18
s	115	01110011	163	73	SC19
t	116	01110100	164	74	SC20
u	117	01110101	165	75	SC21
v	118	01110110	166	76	SC22
w	119	01110111	167	77	SC23
x	120	01111000	170	78	SC24
y	121	01111001	171	79	SC25
z	122	01111010	172	7A	SC26
{	123	01111011	173	7B	SC27
|	124	01111100	174	7C	SC28
}	125	01111101	175	7D	SC29
~	126	01111110	176	7E	SC30
DEL	127	01111111	177	7F	SC31

European Character Codes (128-255)

ASCII Char.	EQUIVALENT FORMS		ASCII Char.	EQUIVALENT FORMS		ASCII Char.	EQUIVALENT FORMS		ASCII Char.	EQUIVALENT FORMS	
	Dec	Binary		Dec	Binary		Dec	Binary		Dec	Binary
NOTE	128	10000000	HF	160	10100000	â	192	11000000	HF	224	11100000
NOTE	129	10000001	HF	161	10100001	ê	193	11000001	HF	225	11100001
NOTE	130	10000010	HF	162	10100010	ô	194	11000010	HF	226	11100010
NOTE	131	10000011	HF	163	10100011	û	195	11000011	HF	227	11100011
NOTE	132	10000100	HF	164	10100100	á	196	11000100	HF	228	11100100
NOTE	133	10000101	HF	165	10100101	é	197	11000101	HF	229	11100101
NOTE	134	10000110	HF	166	10100110	ó	198	11000110	HF	230	11100110
NOTE	135	10000111	HF	167	10100111	ú	199	11000111	HF	231	11100111
NOTE	136	10001000	´	168	10101000	à	200	11001000	HF	232	11101000
NOTE	137	10001001	`	169	10101001	è	201	11001001	HF	233	11101001
NOTE	138	10001010	ˆ	170	10101010	ò	202	11001010	HF	234	11101010
NOTE	139	10001011	¨	171	10101011	ù	203	11001011	HF	235	11101011
NOTE	140	10001100	˜	172	10101100	ä	204	11001100	HF	236	11101100
NOTE	141	10001101	HF	173	10101101	ë	205	11001101	HF	237	11101101
NOTE	142	10001110	HF	174	10101110	ö	206	11001110	HF	238	11101110
NOTE	143	10001111	£	175	10101111	ü	207	11001111	HF	239	11101111
HF	144	10010000	—	176	10110000	Å	208	11010000	HF	240	11110000
HF	145	10010001	HF	177	10110001	î	209	11010001	HF	241	11110001
HF	146	10010010	HF	178	10110010	Ø	210	11010010	HF	242	11110010
HF	147	10010011	°	179	10110011	Æ	211	11010011	HF	243	11110011
HF	148	10010100	HF	180	10110100	å	212	11010100	HF	244	11110100
HF	149	10010101	Ç	181	10110101	í	213	11010101	HF	245	11110101
HF	150	10010110	Ñ	182	10110110	ø	214	11010110	HF	246	11110110
HF	151	10010111	ñ	183	10110111	æ	215	11010111	HF	247	11110111
HF	152	10011000	¡	184	10111000	Ä	216	11011000	HF	248	11111000
HF	153	10011001	¿	185	10111001	ì	217	11011001	HF	249	11111001
HF	154	10011010	¤	186	10111010	Ö	218	11011010	HF	250	11111010
HF	155	10011011	£	187	10111011	Ü	219	11011011	HF	251	11111011
HF	156	10011100	HF	188	10111100	É	220	11011100	HF	252	11111100
HF	157	10011101	§	189	10111101	ï	221	11011101	HF	253	11111101
HF	158	10011110	HF	190	10111110	ß	222	11011110	HF	254	11111110
HF	159	10011111	HF	191	10111111	HF	223	11011111	█	255	11111111

Error Messages

Appendix G

This appendix presents information on errors that may occur when you are using HP TechWriter. Most of the errors that occur in the HP TechWriter programs have self-explanatory messages displayed when they occur. If unusual errors occur, or if the program designer thought that you might need very explicit information about the error, the error will be reported with a codeword (such as *Escapecode*) and an error number. You can look up the error number in the list of errors for that codeword. For example, suppose your disc broke during a write. You would get an error that read: `ERROR, Ioresult=20`. You would look up error number 20 in the *I/O-Related Errors – "Ioresult"* section of this appendix and find the error "Hardware/media failure."

The first section of error messages contains a general discussion about the types of errors that might occur when you are entering or exiting the Editor. The other sections are lists of codeword type errors. Their names include the name of the codeword for the section in quotation marks.

In the event of a non-recoverable error in the Editor, the Editor will attempt to save the contents of the file in memory in the system workfile.

The following sections are included in this appendix:

- I/O Errors Entering or Exiting the Editor
- Operating System Run Time Errors – "Escapecode"
- Input/Output Related Errors – "Ioresult"
- I/O Library Errors – "Ioerror"
- Graphics Errors – "Graphicserror"

I/O Errors Entering or Exiting the Editor

There are two general types of errors that can occur when entering the Editor. The first type of error is generated by the system when it cannot find the volume or file which you specified. The solution to this is to make sure that the proper volume is on-line. This type of error also occurs when a workfile exists but the Editor cannot find it because the medium containing that file is no longer on-line. When the Editor encounters this situation, it informs you that the workfile has been "lost" and then prompts you for a filename.

The second type of error possible while entering the Editor is a memory overflow condition. This happens only if the text file being read was created on a machine with more memory than the machine currently being used. Note that this condition is also met if you use the peRmanent-load command to load something into memory that was not there when you created the text file. (See the *HP TechWriter Environment* chapter of this manual for more information on peRmloading.) Your machine now has less available memory so the space for text files is smaller.

When a memory overflow occurs while reading in the file, the Editor lets you continue the entry process even though the entire file has not been read into memory. However, upon exiting the Editor, the Save and Archive options are no longer available. This safeguard keeps you from accidentally overwriting your original file.

When exiting the Editor, a number of different errors are possible. If the Editor detects an error while writing the contents of the text in the computer's memory to a mass storage medium, it will display a self-explanatory error message.

For error message explanations, see whichever of the following sections is appropriate.

Operating System Run Time Errors – "Escapecode"

If the codeword in an error message is `Escapecode`, look up the error number in this section.

Error	Meaning
0	Normal termination.
−1	Abnormal termination.
−2	Not enough memory.
−3	reference to `NIL` pointer.
−4	Integer overflow.
−5	Divide by zero.
−6	Real math overflow. The number was too large.
−7	Real math underflow. The number was too small.
−8	Value range error.
−9	Case value range error.
−10	Reserved
−11	CPU word access to odd address.
−12	CPU bus error.
−13	Illegal CPU instruction.
−14	CPU privilege violation.
−15	Bad argument – `SIN/COS`.
−16	Bad argument – Natural Log.
−17	Bad argument – `SQRT` (Square root).
−18	Bad argument – real/BCD conversion.
−19	Bad argument – BCD/real conversion.
−20	Stopped by user.
−21	Unassigned CPU trap.
−22	Reserved
−23	Reserved
−24	Macro parameter not `0..9` or `a..z`
−25	Undefined macro parameter.
−26	Error in I/O subsystem.
−27	Graphics error.

Input/Output Related Errors – "Ioresult"

If the codeword in an error message is `Ioresult`, look up the error number in this section.

Error	Meaning
1	Parity (CRC) incorrect. I/O driver will normally do several retries.
2	Illegal unit number. Unit numbers from 1 thru 50 are legal.
3	Illegal I/O request. Such as an attempt to read from a write-only device (printer).
4	Device timeout. Device accessed is not responding.
5	Volume went off-line. A device error detected.
6	File lost in directory. You can have a permanent and temporary file with the same name in a directory. If you purge the "file," you can end up purging both files by mistake.
7	Bad file name.
8	No room on volume. File larger than empty space in volume.
9	Volume not found. Did you spell the name correctly?
10	File not found. Did you spell the name correctly?
11	Duplicate directory entry. One volume cannot have two permanent files with the same name.
12	File already open. You can't open a file that is currently open.
13	File not open. You can't close a closed file.
14	Bad input format. You entered an alpha character when an integer was expected.
15	Disc block out of range. You are trying to read or write to a disc sector larger or smaller than the disc contains.
16	Device absent or inaccessible.
17	Media initialization failed.
18	Media is write protected.
19	Unexpected interrupt.
20	Hardware/media failure.
21	Reserved.
22	DMA absent or unavailable.
23	File size not compatible with type.
24	File not opened for reading.
25	File not opened for writing.

26	File not opened for direct access.
27	No room in directory.
28	String subscript out of range.
29	Bad file close string parameter.
30	Attempt to read past end-of-file mark.
31	Media not initialized.
32	Block not found.
33	Device not ready.
34	Media absent.
35	No directory on volume.
36	File time illegal or does not match request.
37	Parameter illegal or out of range.
38	File cannot be extended.

I/O Library Errors – "Ioerror"

If the codeword in an error message is `Ioerror`, look up the error number in this section.

Error	Meaning
0	No error.
1	No card at select code.
2	Interface should be HP-IB.
3	Not active controller.
4	Should be device address, not select code.
5	No space left in buffer.
6	No data left in buffer.
7	Improper transfer attempted.
8	The select code is busy.
9	The buffer is busy.
10	Improper transfer count.
11	Bad timeout value.
12	No driver for this card.
13	No DMA.
14	Word operations not allowed.
15	Not addressed as talker.
16	Not addressed as listener.
17	A timeout has occurred.
18	Not system controller.
19	Bad status or control.
20	Bad set/clear/test operation.
21	Interface card is dead.
22	End/eod has occurred.
23	Miscellaneous – value of parameter error.
255	Used for `ioe_isc` within I/O errors.
306	Dc interface failure.

313	USART receive buffer overflow.
314	Receive buffer overflow.
315	Missing clock.
316	CTS false too long.
317	Lost carrier disconnect.
318	No activity disconnect.
319	Connection not established.
325	Bad data bits/par combination.
326	Bad status/control register.
327	Control value out of range.

Graphics Errors – "Graphicserror"

If the codeword in an error message is `Graphicserror`, look up the error number in this section.

Error	Meaning
0	No errors since the last call to `graphicserror` or since the last call to `init_graphics`.
1	The graphics system is not initialized.
2	The graphics display is not enabled.
3	The locator device is not enabled.
4	ECHO value requires a graphic display to be enabled.
5	The graphics system is already enabled.
6	Illegal aspect ratio specified.
7	Illegal parameters specified.
8	The parameters specified are outside the physical display limits.
9	The parameters specified are outside the limits of the window.
10	The logical locator and the logical display use the same physical device. The logical locator limits can not be redefined explicitly. They must correspond to the logical view surface limits.
11	The parameters specified are outside the current virtual coordinate system boundary.
12	The escape function requested is not supported by the graphics display device.
13	The parameters specified are outside of the physical locator limits.

Glossary

This glossary contains words that are often used by people using computers and text editors to create documents. The definitions are correct in the context of running the HP TechWriter system or the Pascal Operating System, but may not be precisely correct for all types of computers/editors.

In a definition, if a word is written in **boldface** type, it is a word which is described elsewhere in the glossary.

ASCII	ASCII (American Standard Code for Information Interchange) defines the positions of characters within a 256-entry table of values. See the ASCII tables in the Appendix.
Block	A block is the smallest piece of information which can be transferred between a computer and a **mass storage device**. **Mass storage devices** used by Series 200 computers utilize blocks that are 1 byte long (as on **SRM**), or 256 bytes long (as on internal **floppy discs** and **hard discs**).
Boot	"Boot" means to turn a computer on and bring it to a state in which it can respond to commands. It comes from the phrase "to pull oneself up by one's bootstraps." When you insert the `BOOT:` disc into the default disc drive of your computer, and turn the computer on, the computer system boots. After several seconds, the HP TechWriter **Environment** display appears on your screen, prompting you for the date and time.
Byte	A byte is, loosely speaking, a character. It is a unit of data which is often used in quantifying **file** size.
Command Character	The current command character is defined in the HP TechWriter Environment. If it is the first non-blank character of a line, then the line is not margined, and it delimits **paragraphs**.
Cursor	The cursor is the blinking underline symbol on the screen. It is used in several ways:

- You indicate a position to the computer by placing the cursor on the screen where an action is to take place; for example, you must tell the Editor where to insert or delete something.
- The computer indicates a position to you; for example, when the Editor has found a string.
- When you entering characters the cursor indicates where the next character you type will be placed.

During an editing session with the HP TechWriter Editor, the cursor can (usually) be moved up, down, left and right by pressing ▲, ▼, ◀, and ▶, respectively.

Default Volume	The default volume is the **volume** which is the assumed location of a **file** unless another location is explicitly specified. It may be set either from the `Whatfiles` command from the **Environment** or the `Prefix` command from the **Filer**.
Delete	In the context of a text **editor**, delete specifically means to remove text, and *not* merely to replace it with blanks. In the HP TechWriter **Editor**, the command which does deletion is invoked by pressing (D), (Delete line)(Delete char).
Directory	The directory of a **mass storage medium** is like a telephone directory; it tells the location of and/or how to access **files**. The HP TechWriter Filer directory contains the: file name, file size in **blocks**, location of the first **block** in the file, file type, date created, and date last modified.
Document Command	A document command is a text line whose first two non-blank characters are two adjacent occurrences of the **command character**. In this manual, document commands usually assume the caret (^) is the command character, but you may use any character you wish.
Document Mode	The HP TechWriter **Editor** is in Document Mode if and only if `Auto indent` is `False` and `Filling` is `True`. In this mode word-wrap occurs while inserting text, and **paragraphs** are automatically formatted. Compare **Program Mode**.
Editor	The HP TechWriter Editor is a program which allows you to create, modify, **store** and **retrieve** information. The information being edited by the Editor is in the computer's **memory** until you exit (or stop) the Editor.
Environment	The HP TechWriter Environment is a small operating system from which all the HP TechWriter programs can be accessed. When the computer is **boot**ed, the Environment program is started to allow you to access the HP TechWriter programs.
File	A file is a collection of information which is stored on a **mass storage medium**. A file has associated with it a name, creation date, type, and size in **blocks**, which are stored in a **directory**. When you tell the **Editor** or the **Filer** to do some operation on a file, the computer looks at the directory of the specified **volume**, to see if the file exists. If it does and the operation makes sense, then the operation is executed.
Filer	The Filer is a program which allows you to do many **file**-related and **volume**-related operations, including listing, copying and renaming files and volumes.
File Pool	The file pool is the list of files to be printed in the **Lister** program.
Floppy Disc	A floppy disc is a **mass storage medium** which is composed of a thin plastic circle contained in a protective jacket. The disc spins inside the protective jacket, and a hole in the jacket allows the read/write head of the disc drive to read or write information on the disc. Compare **hard disc**.

Hard Disc	A hard disc is similar to a **floppy disc**, but the disc itself is more reliable, faster to access, has a larger capacity, and is more expensive.
Insert	In the context of the Editor, insert means to create a "hole" in the text, and fill it with "new" text. No existing text is overwritten. When the hole is created the text which is already there is "pushed aside" to make room for the insertion data.
Lister	The Lister is an HP TechWriter program which can take up to fifty files and list them as a single large document. The list of file names, as well as the listing parameters, can be defined, modified, **stored** and **retrieved**.
Mass Storage	See **Mass Storage Medium**.
Mass Storage Device	A mass storage device is a machine which writes information to and/or reads information from a **mass storage medium**. The most common mass storage devices are disc drives, which are devices that read from and write to **floppy discs** or **hard discs**.
Mass Storage Medium	A mass storage medium is the object on which data is actually **stored**, as opposed to the **mass storage device**, which places the information there. For example, a **floppy disc** is a mass storage medium, and the *disc drive* is the **mass storage device**.
Medium	See **Mass Storage Medium**.
Memory	Memory refers to that part of a computer which "remembers" information, also called **RAM** in the context of the HP TechWriter programs.
On-Line	A **mass storage medium** is said to be on-line if it is immediately available (loaded and ready) to the system.
Paragraph	A paragraph is a section of text delimited by any combination of the following: beginning or end of **file**, blank line, or a line whose first non-blank character is a **command character**.
peRmload	The act of loading a program so that it remains in memory until you re-**boot** the computer. The advantage of peRmloading is that it speeds up the access time to a program. The disadvantage is that the program consumes its **memory** space, whether you are running the program or not.
Picture Processor	The Picture Processor is an HP TechWriter program which translates **plot files** into a form which the HP TechWriter Editor and the HP TechWriter Lister can read.
Plot File	A plot file is a file containing the instructions which would have been sent to a plotter, but were detoured into a file. Plot files can be converted by the HP TechWriter Picture Processor into a form that can be used by the **Editor** and the **Lister**.

Program Mode	The HP TechWriter **Editor** is in Program Mode if `Auto indent` is `True` and `Filling` is `False`. In this mode word-wrap does not occur and paragraphs are not formatted - text remains exactly as originally typed. Compare **Document Mode**.
RAM	RAM (random-access **memory**) is the computer's volatile **memory** used for read/write operations. Files are copied from a **mass storage medium** to RAM, where they are modified by the **Editor**. To preserve these modifications the data must be copied back to a **mass storage medium**.
Retrieve	Retrieving data is the process of getting a copy of a file from a **mass storage medium**, and entering it into the computer's **memory** in such a way that the **Editor** can operate on it.
SRM	SRM is an acronym which stands for Shared Resource Management. It is a Hewlett-Packard product which allows desktop computers to share large disc drives, printers, and plotters. SRM utilizes a hierarchical directory structure, which permits many levels of logical partitioning and security.
Store	Storing is the process of taking data which exists in **memory**, and placing it for safekeeping onto a **mass storage medium**.
Stream File	A stream file is a text **file** containing characters which, when used with the HP TechWriter Environment's Stream command, are supplied to an executing program as though they were typed on the keyboard.
System Volume	In the HP TechWriter system, the system volume is where **workfiles** are created, and where temporary data is stored while running **stream files**. The system volume can be designated by using the *Whatfiles* command from the Environment.
Volume	A volume is part of a **mass storage medium** set aside as a unit and called by a particular name. A volume may consume the whole **medium** (as in the case of **floppy discs**), or it may consume just part of the **medium** (as in the case of a **hard disc**).
Workfile	A **file** that resides on the **system volume** named WORK.TEXT. The **Editor** automatically **retrieves** this file (if it exists) when it is started. The **Filer** allows you to designate any file as the workfile with the Get command, and store a copy of the workfile with the Save command. The workfile feature is most valuable for the users of the Pascal Operating System when creating, modifying, and compiling computer programs.

Subject Index