CHAPTER 10

INTRODUCTION TO BATCH FILES

Learning Objectives

After completing this chapter you will be able to:

1. Compare and contrast batch and interactive processing.
2. Explain how batch files work.
3. Explain the purpose and function of the REM, ECHO, and PAUSE commands.
4. Explain how to stop or interrupt the batch file process.
5. Explain the function and use of replaceable parameters in batch files.
6. Explain the function of pipes, filters, and redirection in batch files.

Student Outcomes

1. Use Edit to write batch files.
2. Use COPY CON to write batch files.
3. Write and execute a simple batch file.
4. Write a batch file to load an application program.
5. Use the REM, PAUSE, and ECHO commands in batch files.
6. Terminate a batch file while it is executing.
7. Write batch files using replaceable parameters.
8. Write a batch file using pipes, filters, and redirection.

Chapter Overview

You have used many command line commands throughout this textbook. Many of these commands are repeated in the same sequence. If more than one command is needed to execute a program, you have to key in each command at the system prompt. This repetitive, time-consuming process increases the possibility of human error.

A batch file is a text file that contains a series of commands stored in the order the user wants them carried out. It executes a series of commands with a minimum number of keystrokes. Batch files allow you to automate a process and, at the same time, create more powerful commands, which increases productivity.

In this chapter, you will learn to create batch files to automate a sequence of commands, to write and use batch files for complex tasks, to use batch file subcommands, to halt the execution of a batch file, and to write batch files using replaceable parameters. You will also learn how batch files can be used from the desktop.

10.1 Concepts of Batch and Interactive Processing

Operating system commands used at the command line are programs that are executed or run when you key in the command name. If you wish to run more than one command, you need to key in each command at the system prompt. You can, however, customize and automate the sequence of commands by writing a command sequence, called a ***batch file*** or a command file, to be executed with a minimum number of keystrokes. Any command you can enter at the system prompt can be included in a batch file. You can even execute an application program from a batch file. When you string together a sequence of steps in an application program, it is called a "macro," which is conceptually similar to a batch file.

A batch file contains one or more commands. To create this file of commands, you write a text file using Edit, COPY CON, or a text editor such as Notepad. You can also use a word processor providing it has a "Save as text file" option. The file that you write and name will run any command that the operating system can execute. This file *must* have the file extension .BAT if you are using a version of Windows earlier than Windows 2000 Professional. Beginning with Windows 2000 Professional, you may also use the extension .CMD. The file must be an ASCII file. Once you have written this command file, you execute or run it by simply keying in the name of the batch file, just as you key in the name of a command. The operating system reads and executes each line of the batch file, as if you were sitting at the terminal and separately keying in each command line. Once you start running a batch file, your attention or input is not needed until the batch file has finished executing.

Batch files are used for several reasons. They allow you to minimize keystrokes, and they minimize the possibility of errors, as you do not have to key in the commands over and over. Batch files are used to put together a complex sequence of commands and store them under an easily remembered name. They automate any frequent and/or consistent procedures that you always want to do in the same manner, such as backing up critical data to an alternate location. In addition, you can execute application programs by calling them with a batch file.

"Batch" is an old data-processing term. In the early days of computing, work was done by submitting a job (or all the instructions needed to run the job successfully) to a data-processing department, which would run these jobs in *batches*. There was no chance for anyone to interact with the program. The job was run, and the output was delivered. Thus, when you run a batch job, you are running a computer routine without interruption.

Batch jobs are still run today. An example of a batch job would be running a payroll—issuing paychecks. The computer program that calculates and prints paychecks is run without interruption. The output or results are the paychecks. This job can be run at any time. If a company decides that payday will be Friday, the data-processing department can run the payroll Thursday night. If the company decides payday will be Monday, the data-processing department can run the payroll Sunday night. This is ***batch processing***.

Batch processing is in contrast to an interactive mode of data processing. Sometimes called online or real time mode, interactive mode means you are interacting

directly with the computer. An automated teller machine (ATM) that a bank uses so that you can withdraw or deposit money without human intervention is an example of ***interactive processing***. The bank needs instant updating of its records. It cannot wait until next week to run a batch job to find out how much money you have deposited or withdrawn. If you withdraw $100, the bank first wants to be sure that you have $100 in your account, and then it wants the $100 subtracted immediately from your balance. You are dealing with the computer in an interactive, real time mode—the data is processed without delay.

In the PC world, you can work in interactive mode, but this usually requires a connection to another computer, over phone lines, DSL, or a cable modem. The Internet allows you to communicate directly with other computers and perform such functions as reviewing airline flight schedules and booking an airline ticket. Although interactive mode can be exciting, most of the time you are working one-on-one with your computer and are not in interactive mode. Hence, the batch mode is the area of emphasis.

10.2 How Batch Files Work

You will be creating and executing batch files in this chapter. By now you should know that data and programs are stored as files, but how does the operating system know the difference between a data file and a program file? As mentioned in previous chapters, it knows the difference based on the file extension. When you key in something at the prompt, the operating system first checks in RAM to compare what you keyed in to the internal table of commands. If it finds a match, the program is executed. If what you keyed in does not match an internal command, the operating system looks on the default drive and directory for the extension .COM, meaning command file, first. Then the operating system looks for the file extension .EXE, meaning executable file (this extension is used for system utility programs and most application software).

If what you keyed in does not match either .COM or .EXE, the operating system looks on the default drive and directory for the file extension .BAT, meaning batch file. If it finds a match, it loads and executes the batch file, one line at a time. It then looks for .CMD, meaning command file. If it finds a match, it loads and executes the command file, one line at a time. If what you keyed in does not match any of the above criteria, it continues to search in your default directory for files with the following extensions: .VBS, .VBE, JS, JSE, WSF and WSH. Table 10.1 lists the search order for extensions.

Extension	Meaning
.COM	Command file
.EXE	Executable file
.BAT	Batch file
.CMD	Command script file
.VB	VBScript file (Visual Basic)
.VBE	VBScript Encoded Script file (Visual Basic)
.JS	JScript file (JavaScript)

.JSE	JScript Encoded Script file (JavaScript)
.WSF	Windows Script file
.WSH	Windows Script Host Settings file

Table 10.1—Search Order for Extensions

If the command interpreter does not find any of these in your default drive and directory, it then searches your search path as set in the PATH statement, in the file extension order listed above. If your file name does not meet any of these criteria, then you see the error message, "*filename* is not recognized as an internal or external command, operable program or batch file."

What if you had files on a disk that had the same file name but three different file extensions, such as CHKDSK.COM, CHKDSK.EXE, and CHKDSK.BAT? How would the operating system know which program to load and execute? Priority rules are followed. The operating system looks for the program with the .COM file extension first and, if found, would never get to the other files. However, if you were more specific and keyed in both the file name *and* the file extension, such as CHKDSK.BAT, the operating system would then execute the file name you specified.

Remember that, since the batch file is a program, either the .BAT file must be on the default drive and directory or the path must be set to the location of the batch file so you may invoke it. Most importantly, each line in a batch file must contain only one command.

10.3 Using Edit to Write Batch Files

To write batch files, you need a mechanism to create an ASCII text file, since that is what a batch file is. You should remember that ASCII is a code used by the operating system to interpret the letters, numbers, and punctuation marks that you key in. In simple terms, if a file is readable with the TYPE command, it is an ASCII text file. You can use a word-processing program to write a batch file if it has a nondocument or text mode. However, most word-processing programs are quite large and take time to load into memory. Most batch files are not very long, nor do they need the features of a word processor. Using a word processor to write a batch file is like using a sledgehammer to kill a fly.

Having a small, simple text editor is so important that the operating system includes one as part of the system utility programs. This is the Command Line editor, called Edit. Edit is simple to use and universal. You will write some batch files using Edit. Remember, Edit is only a tool to create the file; it does not run or execute it. You execute the file when you key in the file name at the system prompt in the Command Line window. Each line in a batch file must contain only one command. A batch file can have any legal file name but the file extension must always be .BAT or .CMD.

If you are in the Windows interface, the text editor is Notepad. Like Edit, Notepad creates text-only files and may be used to write batch files. However, if you are having problems with Windows, you will not have Notepad available to you because you need a graphical user interface to use Notepad. Edit, on the other hand,

can work at the command line. In fact, you will later see that when you create your startup disk, Edit is on the disk, but not Notepad. Thus, in the following activities, you will be using Edit.

10.4 Activity: Writing and Executing a Batch File

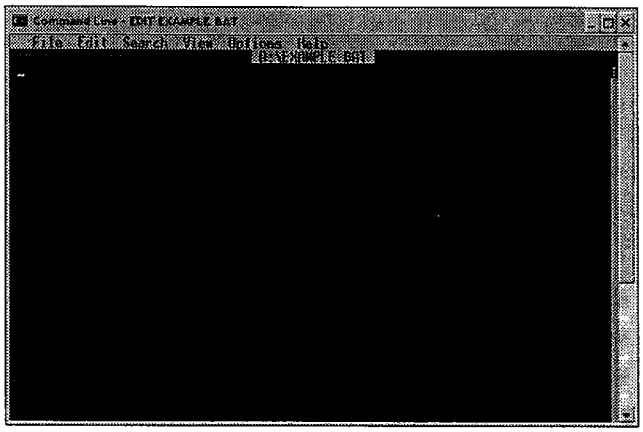
Note 1: The DATA disk is in Drive A. A:\> is displayed.

Note 2: Although you may use a mouse with the MS-DOS editor, the instructions will show the keystroke steps, not the mouse steps.

Note 3: In some systems, the mouse will not work in Edit unless you change the properties of Edit to open Edit in full-screen mode or clear the Quick Edit Mode check box in the Command Line property sheet (Options tab).

Note 4: The amount of space shown as remaining on the disk will vary, depending on the size and placement of the batch files on your disk.

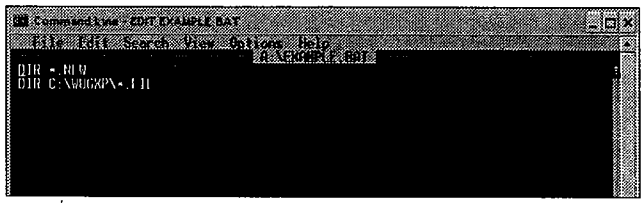
1 Key in the following: `A:\>`**EDIT EXAMPLE.BAT** Enter



WHAT'S HAPPENING? You are now using the Command Line editor. You are going to create a batch file named **EXAMPLE**. The file extension must be **.BAT** or **.CMD**.

2 Key in the following: **DIR *.NEW** Enter

3 Key in the following: **DIR C:\WUGXP*.FIL** Enter

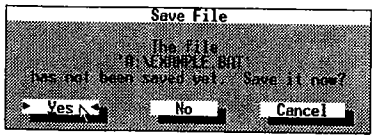


WHAT'S HAPPENING? Look at each line. Each one is a legitimate operating system command that could be keyed in at the prompt. Each command is on a separate line. The first line asks for a listing of all the files on the disk in the default drive that have the file extension **.TXT**. The second line asks for all the files in the WUGXP

directory in the C drive that have the file extension **FIL**. At this point, you have written the batch file. Next, you need to exit Edit and save the file to disk.

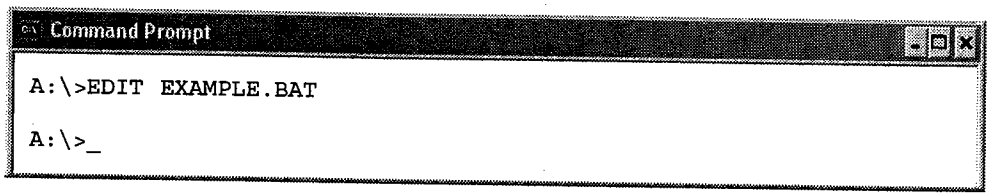
4 Press **Alt** + **F**.

5 Press **X**.



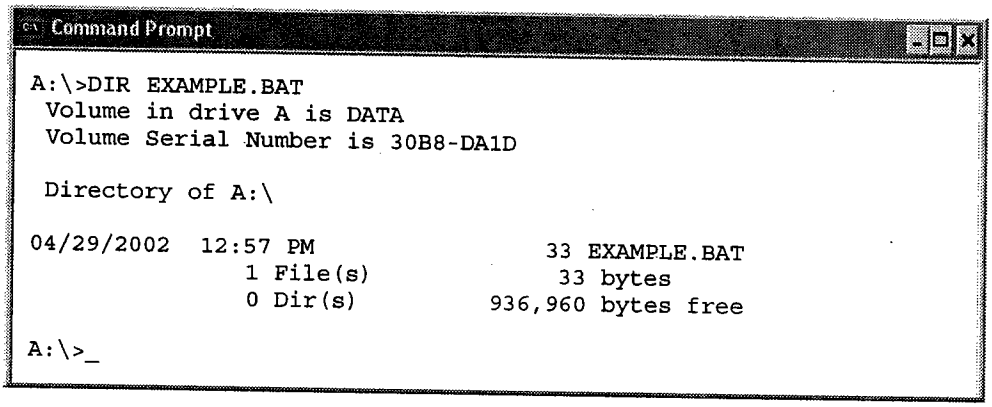
WHAT'S HAPPENING? Since you have not saved the file, Edit reminds you with a dialog box that, if you want this file on the disk, you must save it.

6 Press **Y**.



WHAT'S HAPPENING? You have saved your file, exited Edit, and returned to the system prompt.

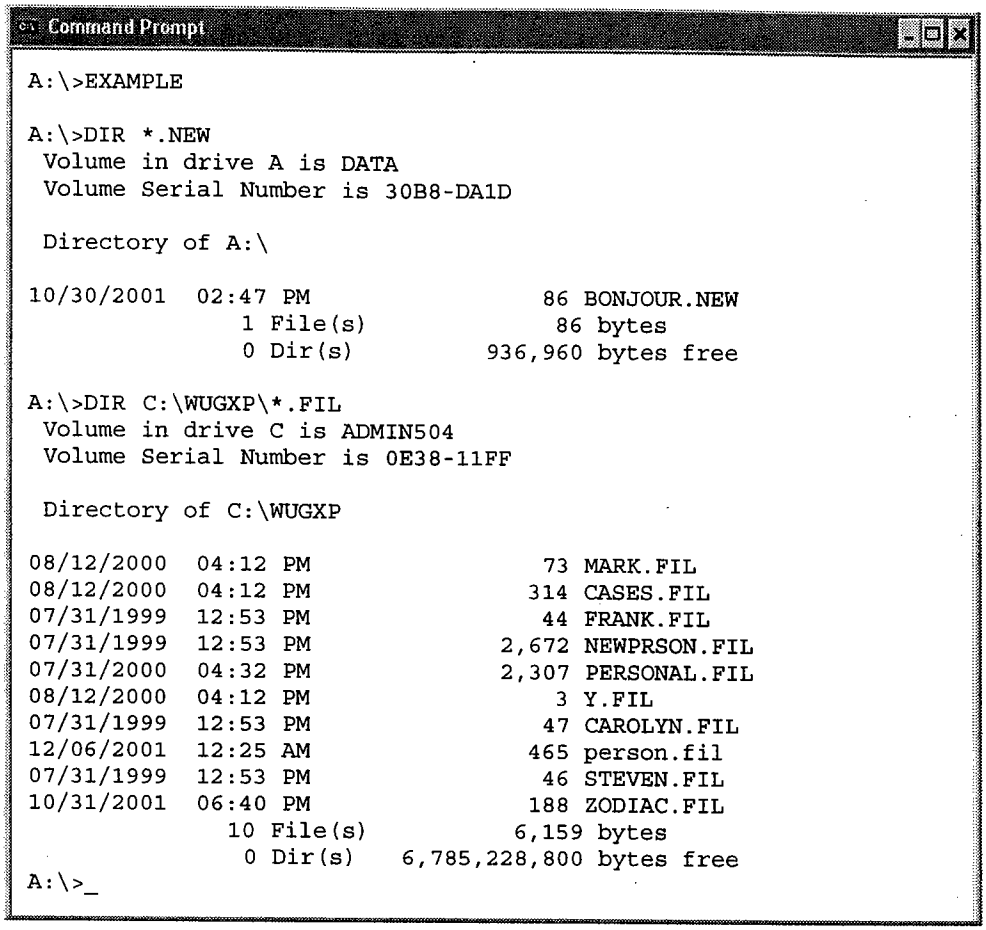
7 Key in the following: `A:\>`**DIR EXAMPLE.BAT** **Enter**



WHAT'S HAPPENING? The **DIR EXAMPLE.BAT** command shows that there is a file on the DATA disk called **EXAMPLE.BAT**. It looks like any other text file.

How do you make the operating system treat it like a program so that you can execute it? You simply key in the name of the file at the prompt. You do not need to key in the extension, just the name. The operating system first looks for a file in its internal table called **EXAMPLE**. It does not find it. It then looks for a file called **EXAMPLE.COM** on the default disk, the DATA disk. No file exists called **EXAMPLE.COM**. Next, it looks for a file on the default disk called **EXAMPLE.EXE**. No file exists called **EXAMPLE.EXE**. It then looks for a file called **EXAMPLE.BAT** on the default disk. It does find a file by this name. It loads it into memory and executes each line, one at a time. Thus, to execute the batch file called **EXAMPLE**, key in the name of the file at the prompt. Watch what happens on the screen after you key in the file name.

8 Key in the following: A:\>**EXAMPLE** Enter



WHAT'S HAPPENING? (*Note:* Part of the display may have scrolled off of your screen.) The operating system read and executed each line of the batch file you wrote, one line at a time. The screen displayed each command line and the results of the command line as it executed. Each line executed as if you had sat in front of the keyboard and keyed in each command individually. You did key in the commands when you wrote the batch file, but you had to key them in only once. The first line was **DIR *.NEW**. When the operating system read that line, it executed it and showed on the screen the file on the DATA disk with the file extension **.NEW**. It then read the next line of the batch file and looked in the root directory of Drive C for any file that had the file extension **.FIL**. It found 10 files with that extension, and displayed their names on the screen. Now that you have written the file **EXAMPLE.BAT**, you can execute this batch file's commands over and over again by keying in **EXAMPLE** at the prompt.

10.5 Writing and Executing a Batch File to Save Keystrokes

The previous example showed you how to write and execute a batch file, but that file is not especially useful. The next batch file to be written will allow you to key in only one keystroke instead of seven. As you know, the command DIR /AD will

quickly show you any subdirectories on the DATA disk. The /A switch means attribute, and the attribute you want displayed is D for directories. This command is composed of seven keystrokes, and you must have the proper parameters. With a batch file, you can do the same task by pressing only one key.

The DIR command has other parameters that are very useful. One of these is O for order. There are many kinds of order you can achieve. One kind that is useful is the arrangement of files by size. The command line would be DIR /OS. The O is for order, and the S is to arrange by size from the smallest to the largest file. If you wanted to reverse the order so that the files would be displayed from the largest to smallest, the command would be DIR /O-S. The O is still for order, but the - is for reverse order, placing smallest files at the end of the listing. The S is for file size. This command would take eight keystrokes. You can reduce it to one.

These batch files you are going to write are very small—one line. It seems like a lot of trouble to load Edit just to accomplish this task. If you would rather not load Edit, you can use the COPY command to write a simple ASCII file. The syntax is:

```
COPY CON filename
```

What you are doing here is copying what you key in (CON) to a file name. CON is the operating system's name for the keyboard/console devices of your computer. You are still following the syntax of the COPY command; it is just that now you are copying from a device—the console (CON)—to a file. Remember that in an early chapter, you copied to a device, the printer (COPY filename PRN). Just as PRN, LPT1, and LPT2 are reserved device names, so is CON.

When you are done keying in text, you must tell the COPY command you are finished. You do this by pressing the F6 key and then the Enter key. This writes the data you keyed in to the file name you specified. This is what you have been doing in your Homework assignments when you have entered data in NAME.FIL. The only problem with COPY CON, as it is informally referred to, is that you cannot correct errors once you press Enter at the end of a command line. Nor can you use COPY CON to correct errors in an existing file. To do that, you need an editor, such as Edit. But nothing is faster than using COPY CON.

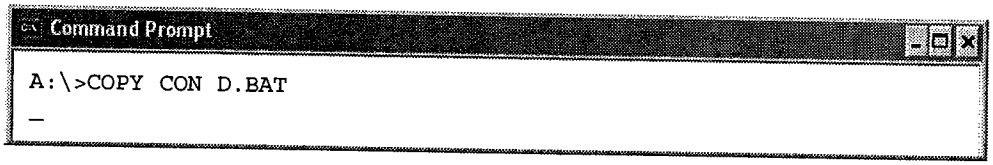
10.6 Activity: Writing and Executing a One-letter Batch File

Note 1: The DATA disk is in Drive A. A:\> is displayed.

Note 2: For these examples, the use of COPY CON will be shown. If you make errors, you can either use COPY CON and key in all the data again or use the MS-DOS editor to correct the errors.

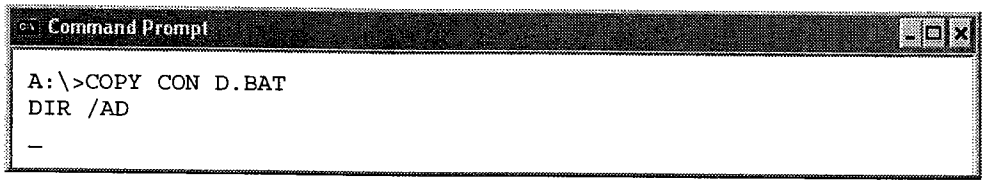
Note 3: In earlier chapters you may have used DOSKEY and the function keys to correct errors. Either of these methods will work with COPY CON.

1 Key in the following: A:\>**COPY CON D.BAT** Enter



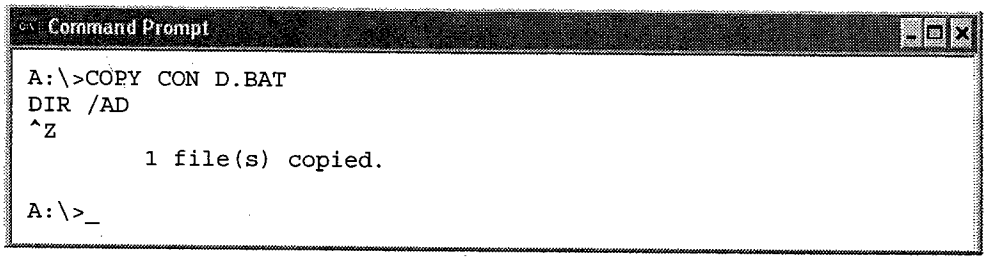
WHAT'S HAPPENING? When you keyed in **COPY CON D.BAT**, you were informing the COPY command that you wanted to make the keyboard the source. The cursor is blinking right below the prompt, and the screen is blank.

2 Key in the following: **DIR /AD** Enter



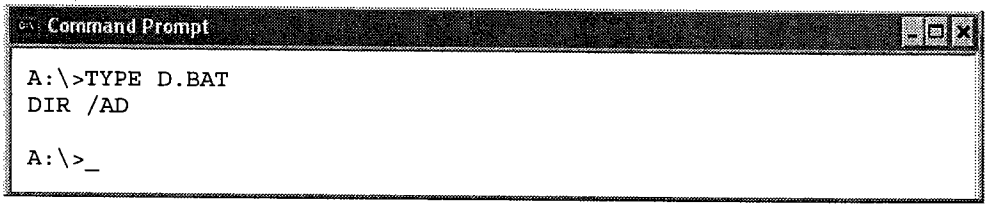
WHAT'S HAPPENING? You have one line. You are finished keying in data and you wish this line to be saved to a file called **D.BAT**. First, however, you must tell COPY you are finished.

3 Press F6 Enter



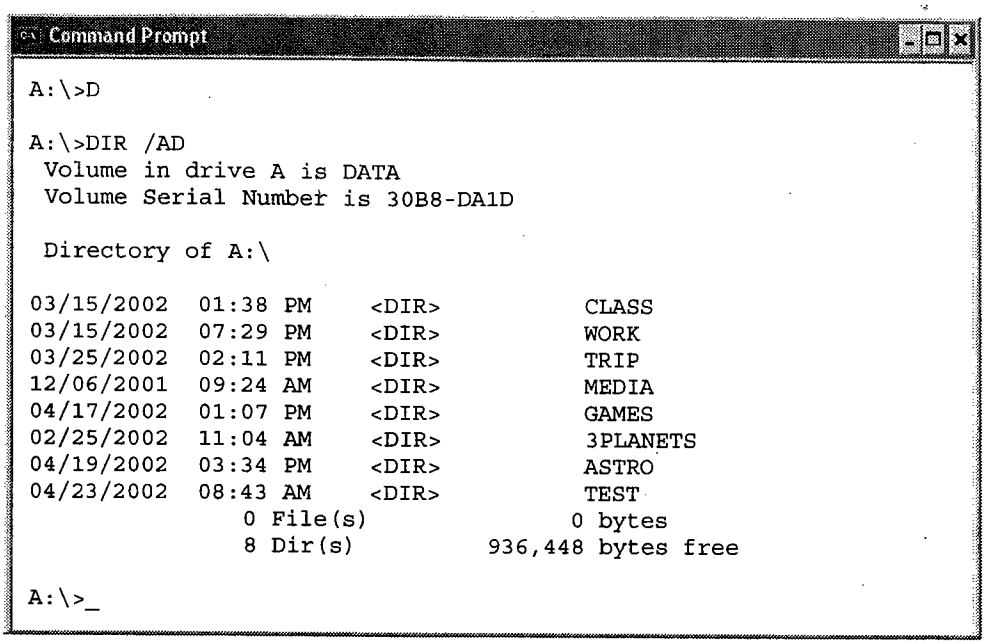
WHAT'S HAPPENING? By pressing F6 and then Enter, you sent a signal to COPY that you were done. The F6 appeared on the screen as **^Z**. Pressing Ctrl + Z will produce the same results as F6. You then got the message "1 file(s) copied" and were returned to the system level.

4 Key in the following: A:\>**TYPE D.BAT** Enter



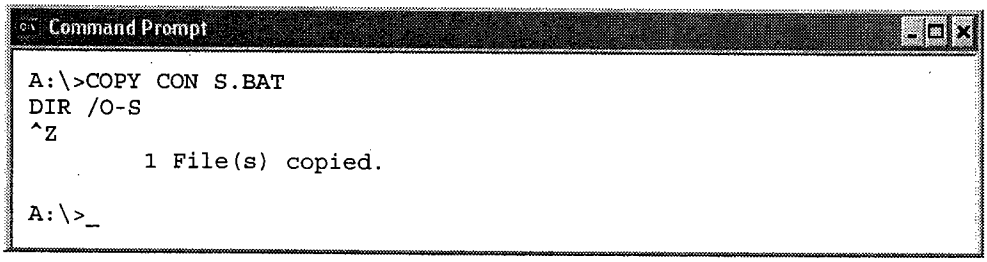
WHAT'S HAPPENING? You wrote a one-line batch file named **D.BAT** with COPY CON and saved the file **D.BAT** to the disk. Once you returned to the system prompt, you displayed the contents of **D.BAT** with the TYPE command. The fact that you could display this file with the TYPE command is another indication that it is indeed an ASCII file. All COPY CON did was allow you to create the file, and TYPE merely displayed what is inside the file. To execute the file, you must key in the file name. Now, whenever you want to see the subdirectories on the DATA disk in Drive A, you only have to key in one letter to execute this command.

5 Key in the following: A:\>**D** Enter



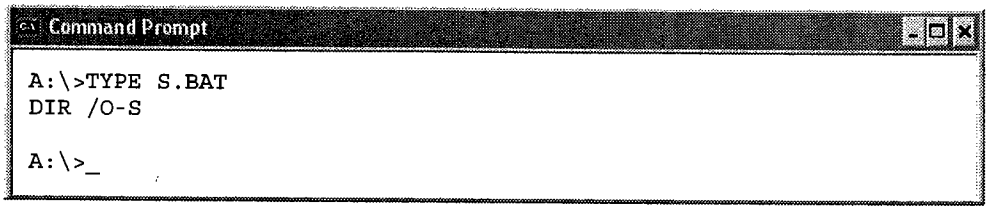
WHAT'S HAPPENING? Your display may vary based on what subdirectories are on the DATA disk and in what order they were created. As you can see, you set up a command sequence in a batch file called **D.BAT**. You can run this batch file whenever the need arises, simply by keying in the name of the batch file at the system prompt. You can also display the files by size, with the smallest file at the end of the list.

6 Key in the following: `A:\>`**COPY CON S.BAT** Enter
DIR /O-S Enter
F6 Enter



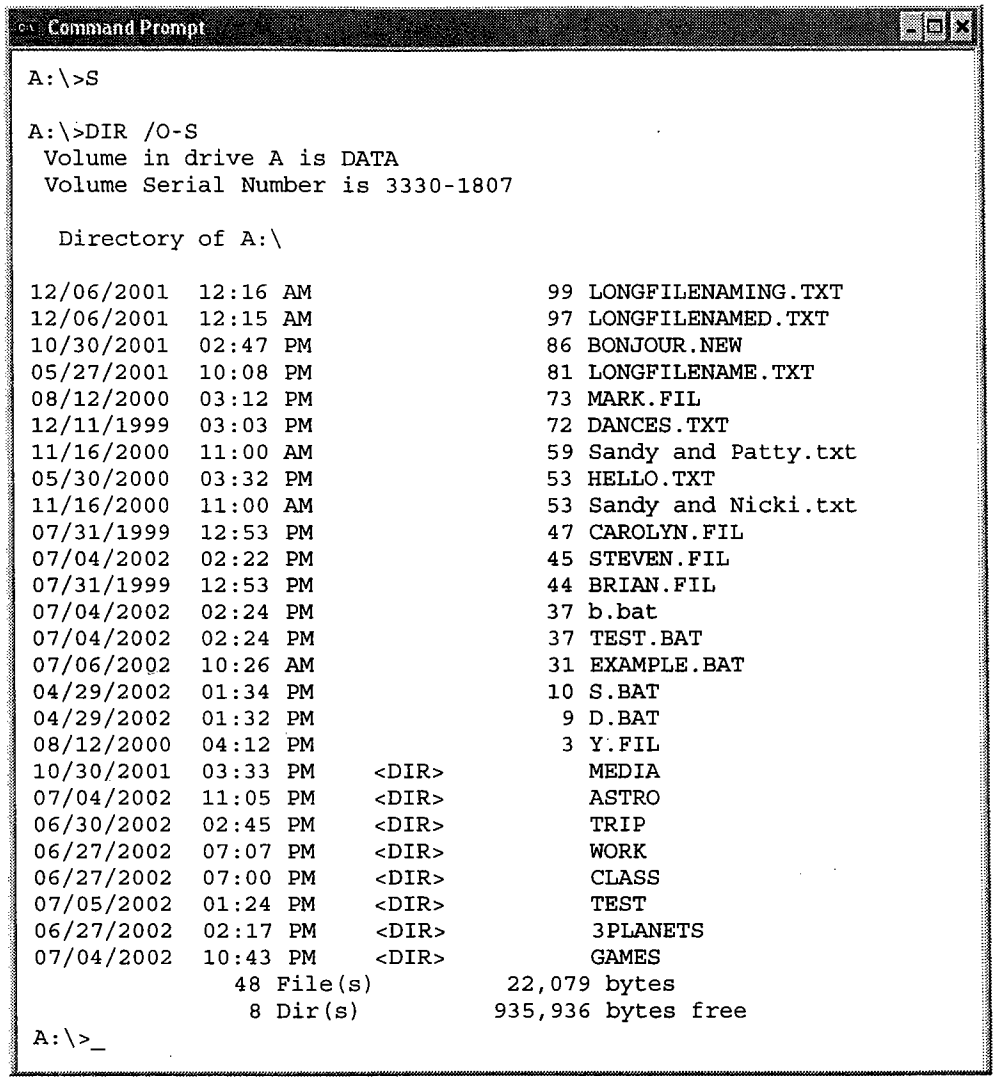
WHAT'S HAPPENING? You have written another simple one-line batch file and saved it to the default directory.

7 Key in the following: `A:\>`**TYPE S.BAT** Enter



WHAT'S HAPPENING? After saving the file to disk, you looked at its contents with the TYPE command. To execute the batch file, you must key in the batch file name (**S**) at the system prompt.

8 Key in the following: `A:\>`**S** Enter



WHAT'S HAPPENING? (The graphic represents the top and bottom of what you will see scroll by on your screen.) The files are listed by size, and all the subdirectories are grouped at the bottom of the display. Because directories have no size, they are listed last as the smallest files.

10.7 Using Batch Files to Alter Your Command Line Environment

Today, in the Windows environment, you typically open a command line session by clicking on the icon on the Start Menu. The window can be opened from a short cut as well. The short cut can then be altered to run in a customized way.

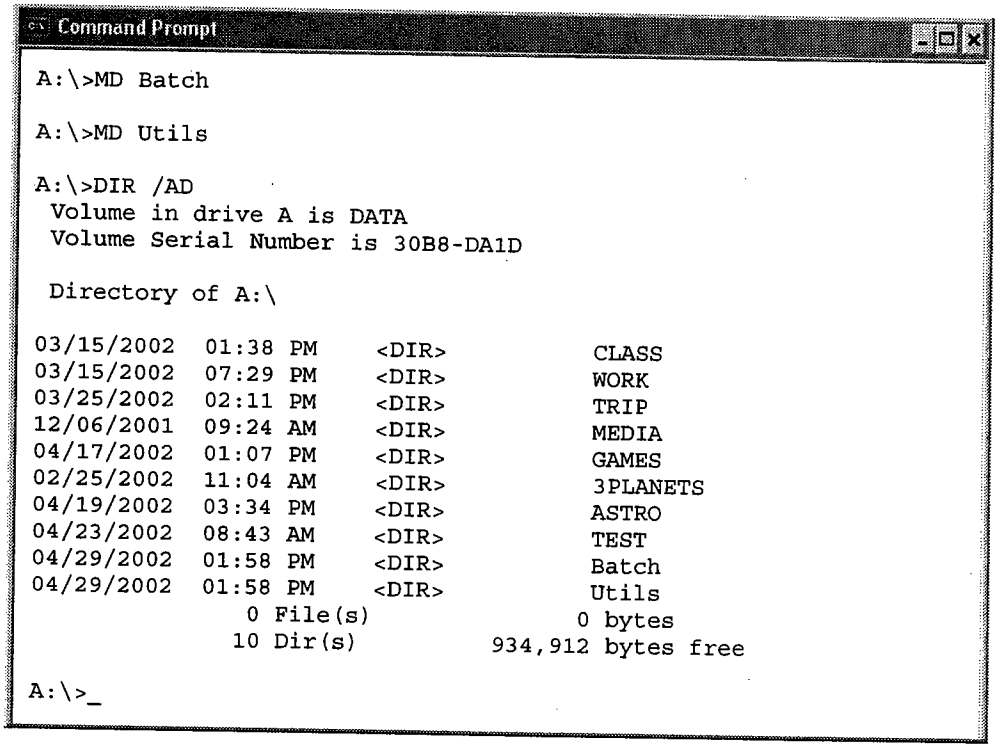
In Chapter 8, disk organization was discussed. It was suggested that, on your own system, you might create a subdirectory called Batch to hold your batch files, and Utils to hold utility files. These two user-created directories will not be part of the normal path in a command line window. If you have batch files that you like to use when in a command line session, it would be helpful if the default path included these two directories.

You will first create two subdirectories on the DATA disk, and then add them to the path for use in your command line sessions. Note: To be truly useful, the directories would be on your own system on the hard drive. Using the DATA disk to do this could, in the real world, present a serious problem, in that the DATA disk that contained the two subdirectories would have to be in the A drive each and every time you used your short cut to begin a command line session.

10.8 Activity: Creating a Batch File to Alter the Command Line Session Environment

Note: The DATA disk is in Drive A. A:\> is displayed.

1. Key in the following: `A:\>`**MD Batch** Enter
2. Key in the following: `A:\>`**MD Utils** Enter
3. Key in the following: `A:\>`**DIR /AD** Enter



WHAT'S HAPPENING? You have created two subdirectories on the DATA disk and used the DIR command to verify their existence. Notice that the operating system remembers the case you used when you created the directories.

You would like the directory names to be in upper case. You can use the REN command to change the case.

4 Key in the following: `A:\>`**REN Batch BATCH** Enter

5 Key in the following: `A:\>`**REN Utils UTILS** Enter

6 Key in the following: `A:\>`**DIR /AD** Enter

```
Command Prompt
A:\>REN Batch BATCH

A:\>REN Utils UTILS

A:\>DIR /AD
 Volume in drive A is DATA
 Volume Serial Number is 30B8-DA1D

 Directory of A:\

03/15/2002  01:38 PM    <DIR>          CLASS
03/15/2002  07:29 PM    <DIR>          WORK
03/25/2002  02:11 PM    <DIR>          TRIP
12/06/2001  09:24 AM    <DIR>          MEDIA
04/17/2002  01:07 PM    <DIR>          GAMES
02/25/2002  11:04 AM    <DIR>          3PLANETS
04/19/2002  03:34 PM    <DIR>          ASTRO
04/23/2002  08:43 AM    <DIR>          TEST
04/29/2002  01:58 PM    <DIR>          BATCH
04/29/2002  01:58 PM    <DIR>          UTILS
               0 File(s)              0 bytes
              10 Dir(s)         934,912 bytes free

A:\>_
```

WHAT'S HAPPENING? You have used the REN command to change the case of the directory names.

7 Key in the following: `A:\>`**EDIT A:\BATCH\SETPATH.BAT** Enter

8 In the Edit screen, Key in the following:
PATH = %PATH%;A:\BATCH;A:\UTILS

9 Press Alt + **F**.

10 Press **X**.

11 Press the **Y** key to save the file.

```
Command Prompt
A:\>EDIT A:\BATCH\SETPATH.BAT

A:\>_
```

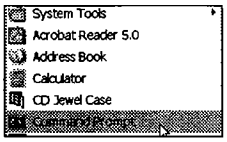
WHAT'S HAPPENING? You have completed creating the SETPATH.BAT file, and returned to the command line window.

12 Close the Command line window.

Note: If you still have the shortcut to the command line window on your desktop, do not do Steps 13 through 17; go directly to Step 18.

13 Point to **Programs**. Point to **Accessories**.

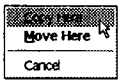
14 Point to the Command Prompt icon.



WHAT'S HAPPENING? You have displayed the Icon to start a command line session.

15 Hold down the right mouse button and drag the command icon onto the desktop.

16 Release the right mouse button.



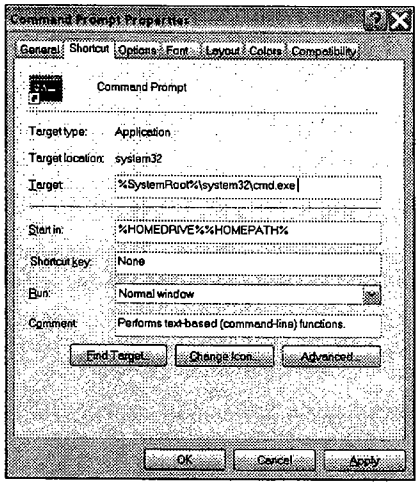
WHAT'S HAPPENING? You see a shortcut menu appear.

17 Click **Copy Here**.

WHAT'S HAPPENING? You now have a short cut to open a command line session on the desk top.

18 Right-click the shortcut icon, and click **Properties**.

19 Click the **Shortcut** tab.



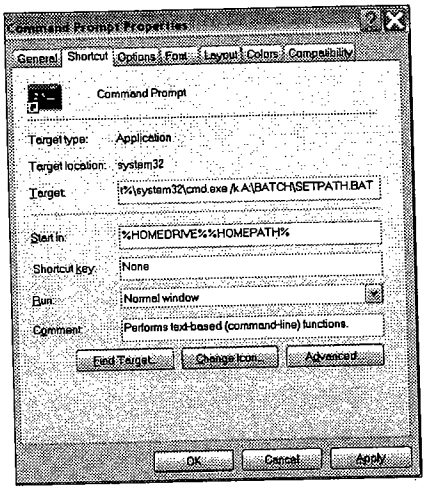
WHAT'S HAPPENING? You have opened the property sheet for the shortcut to the CMD.EXE program. Notice entry in the Target text box. When you click this Icon, the operating system goes to the root of this system (%SystemRoot%), to the system32 subdirectory, and runs the program file cmd.exe. Also, notice the Start In text box. In this example, the computer is part of a Domain network, which assigns a particular home directory to the user. The line %HOMEDRIVE%%HOMEPATH% causes the initial default directory to be the directory assigned by the network administrator to the user. On this particular computer, that will be a directory on the server addressed as G.

In your own home environment, you could change this to say C:\ or C:\DATA or whatever directory you wished to be the default directory when you open a command session.

20 Click in the Target text box. Press the End key.

21 Press the spacebar once.

22 Key in the following: **/k A:\BATCH\SETPATH.BAT**

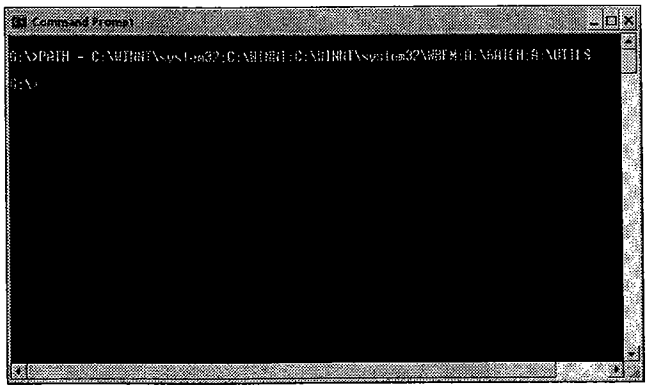


WHAT'S HAPPENING? You have given additional instructions to the Target command line. You have told the operating system to go to the specified drive, run cmd.exe to open a command line session, and then to run the file **SETPATH.BAT** residing in the specified A:\BATCH directory.

23 Click **Apply**.

24 Click **OK**.

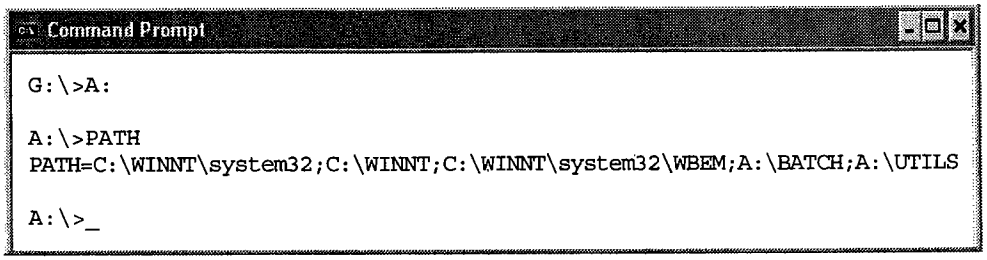
25 Double-click the shortcut to the Command line icon on the desktop.



WHAT'S HAPPENING? As you can see, the batch file executed. But, did it work?

26 Key in the following from the default prompt: **A:** Enter

27 Key in the following: `A:\>`**PATH** Enter

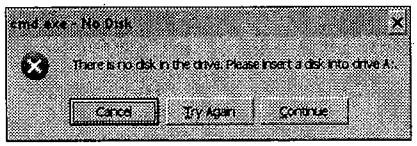


WHAT'S HAPPENING? As you can see by the path returned, the batch file did indeed work. Each time you use this shortcut icon to open a command line session, the path will include the specified directories on the A drive. Once again, this will only be helpful if this particular disk is in the A drive.

28 Close the command line window.

29 Remove the DATA disk from the A drive.

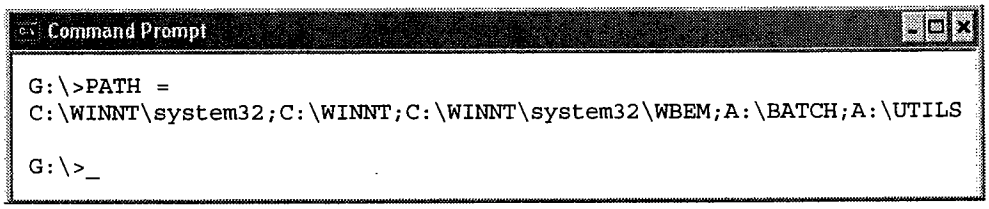
30 Double-click the shortcut icon.



WHAT'S HAPPENING? The message box tells you that it is running the program cmd.exe and that program is reporting there is No Disk in the drive.

31 Insert the DATA disk into the A drive.

32 Click **Try Again**.



WHAT'S HAPPENING? The operating system found the specified disk and directories, and executed the batch file requested.

With the /k parameter, you can include a batch file with any number of commands that you would like to execute each time you open a command line session. Once again, the use of this technique on a floppy has limited value, but on your own system, referencing directories on the hard drive, it can be very valuable.

33 Close all open windows.

34 Drag the shortcut to the Recycle bin to delete it.

10.9 Writing a Batch File to Load an Application Program

Previously, in order to execute the BOG game, you had to take three separate steps. First you had to change to the directory where the program file was located. Second, you had to load the program, BOG.EXE. Third, after you exited the game, you returned to the root directory. A batch file is an ideal place to put all of these commands. In the following activity, you will run the BOG game from its location on the hard drive. Remember, if your WUGXP directory is somewhere other than C:\, substitute that location in your batch file.

10.10 Activity: Writing a Batch File to Execute the BOG Game

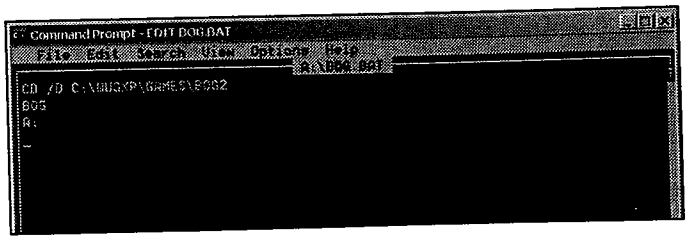
Note: You may use any text editor you wish for creating the batch files. You may use COPY CON, but when you have more than one line, using an editor is easier. Remember, you cannot edit lines when you use COPY CON. The operating system's Edit program will be used in this text. Instructions for keyboard use will be shown, but if you prefer using a mouse, do so.

1 Open a command line window.

2 Change directories to the root of the A drive.

3 Key in the following: `A:\>`**EDIT BOG.BAT** Enter

4 Key in the following: **CD /D C:\WUGXP\GAMES\BOG2** Enter
BOG Enter
A: Enter



WHAT'S HAPPENING? You have just written a batch file to load the BOG application program. You can give the file the name **BOG.BAT**, because it is in the root directory

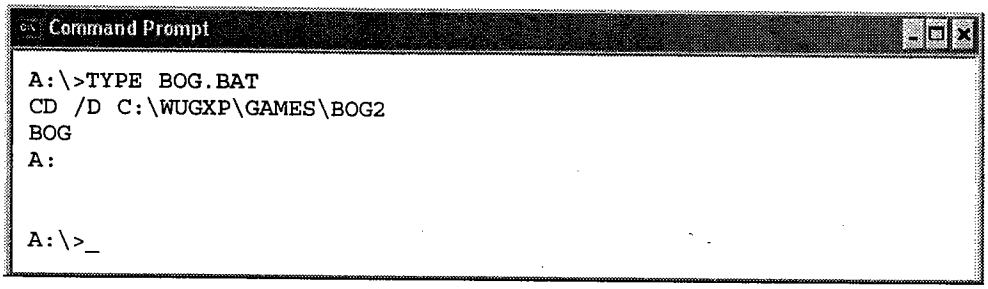
of the DATA disk and **BOG.EXE** is in a subdirectory on the hard drive. The two file names will not conflict, and you will not have to be specific and key in **BOG.BAT**. Furthermore, the first line tells the operating system to change drives and directories. The full path name is necessary in to change to the desired directory, but even if it had not been, you want the batch file to run no matter where you are when you execute it, so commands the require a particular location should always be referenced by their full file specification.

5 Press **Alt** + **F**.

6 Press **X**.

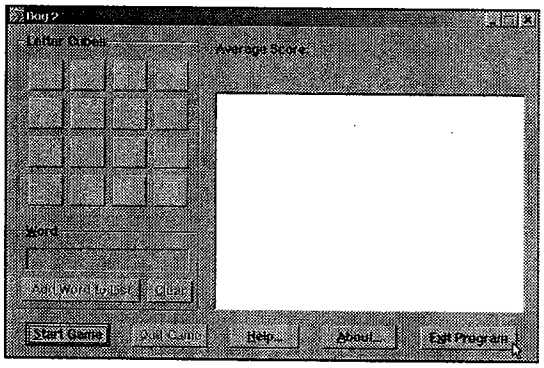
7 Press **Y**.

8 Key in the following: `A:\>`**TYPE BOG.BAT** **Enter**



WHAT'S HAPPENING? You created the batch file **BOG.BAT** in Edit and then returned to the system prompt. Now you can execute this file.

9 Key in the following: `A:\>`**BOG** **Enter**



WHAT'S HAPPENING? When you keyed in BOG at the root directory, the operating system looked for changed directories, looked for and found the BOG program, and executed it. It read the lines in the order they appeared. It read the first line, which said to change the drive and directory to the **C:\WUGXP\GAMES\BOG2** subdirectory. It then read the second line, which told it to look for a program called **BOG.EXE**, and then it loaded BOG. The BOG game then appeared on the screen.

10 Click the **Exit Program** button.

```
Command Prompt
A:\>BOG

A:\>CD /D C:\WUGXP\GAMES\BOG2

C:\WUGXP\GAMES\BOG2>BOG

C:\WUGXP\GAMES\BOG2>A:
A:\>_
```

WHAT'S HAPPENING? You can see on the command line screen how the lines were executed, one at a time. It does not matter if played the game for one minute, one hour, or one entire day. Whenever you exit the program, the command line continues with the execution of the batch file where it last was and simply reads and executes the next line. The operating system finished executing your batch file by changing the drive back to the A drive.

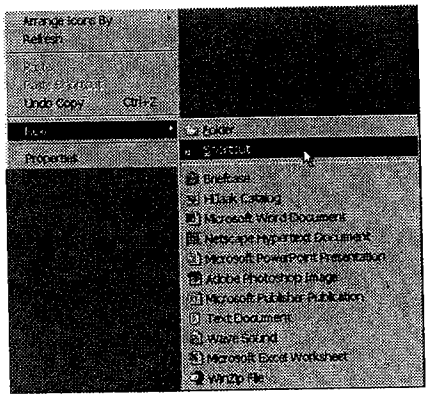
10.11 Creating Shortcuts for Batch Files on the Desktop

Any batch file can be run from the Windows environment. One way to do it is to locate the batch file name in Windows Explorer or My Computer, then double-click the file name. You can also create a shortcut for it and place it on the desktop or in a folder. Again, once it is a shortcut, the shortcut can be clicked to execute the batch file. However, there are things that you can do with the shortcut that you cannot do in the command line interface. You can also change the icon for the shortcut.

10.12 Activity: Creating a Shortcut on the Desktop

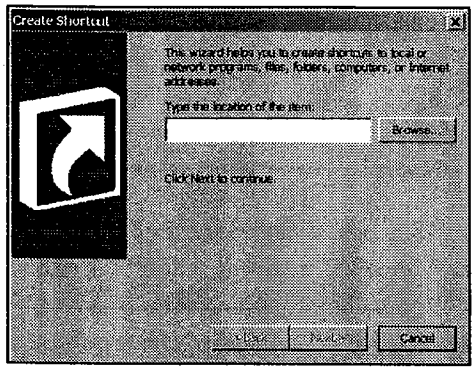
Note: The DATA disk is in Drive A. A:\> is displayed.

1. Key in the following: `A:\>`**EXIT** Enter
2. Right-click in any blank area of the desktop.
3. Point to **New**.



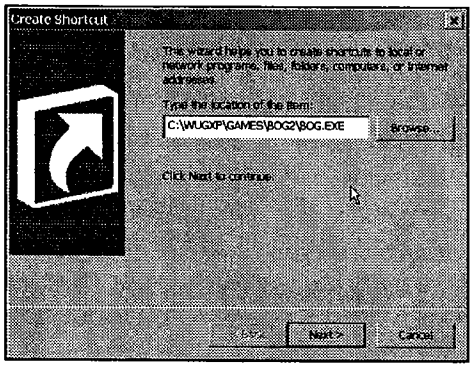
WHAT'S HAPPENING? You opened the shortcut menu for the desktop, and the pop out menu for New.

4 Click **Shortcut**.



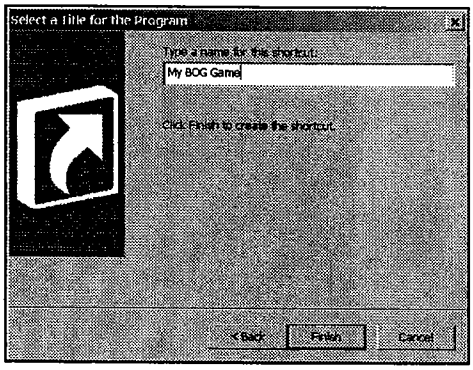
WHAT'S HAPPENING? You started the Create a shortcut wizard.

5 In the location box, key in **C:\WUGXP\GAMES\BOG2\BOG.EXE**.



WHAT'S HAPPENING? Notice that, as you type, the operating system anticipates what you mean. Once you key in the B of BOG, the BOG2 directory was shown below. As you continued to key in the full name, the B of BOG.EXE brought up 4 possibilities. You could have, at this point, double clicked on BOG.EXE and populated the location box in this way. However, in this example, you keyed in the entire file specification.

6 Click **Next**.



WHAT'S HAPPENING? You are asked to supply a name for the shortcut. Once again, Windows XP anticipates what you want. This time, however, the assumption is incorrect.

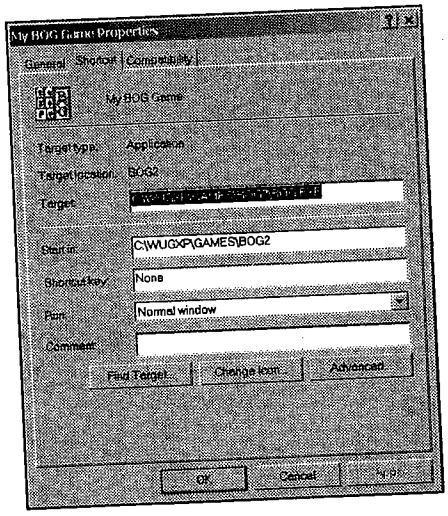
7 In the name box, key in **My BOG Game** as shown in the figure above.

8 Click **Finish**.

WHAT'S HAPPENING? You now have a shortcut to the batch file you created. Notice the Icon. Many executable files written for the Windows environment contain an icon that represents the program. If this program had not contained an icon, you would see the default windows icon. You can change the icon for short cuts..

9 Right-click the shortcut.

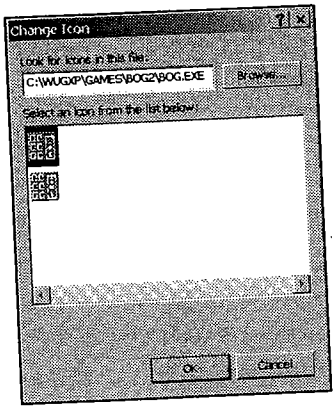
10 Click **Properties**.



WHAT'S HAPPENING? You have opened the property sheet for the shortcut to the BOG program.

11 Click the **Shortcut** tab, if it is not already selected.

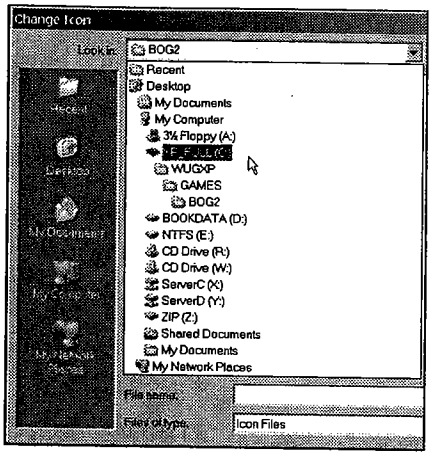
12 Click the **Change Icon** button.



WHAT'S HAPPENING? As you can see, the program file BOG.EXE has its own icons. You want to change this.

13 Click **Browse.**

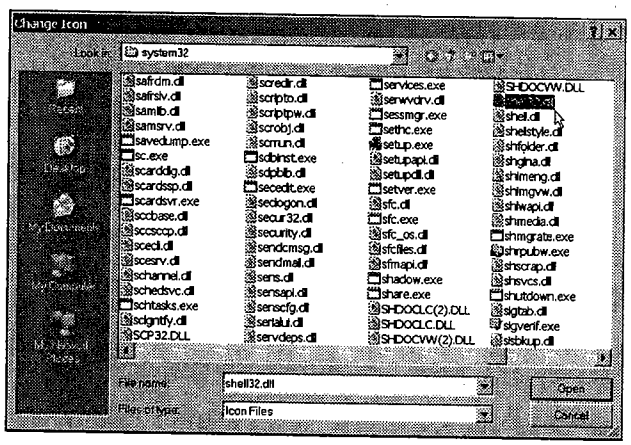
14 Click the down arrow of the **Look for icons in this file:** text box.



15 Click the C drive icon.

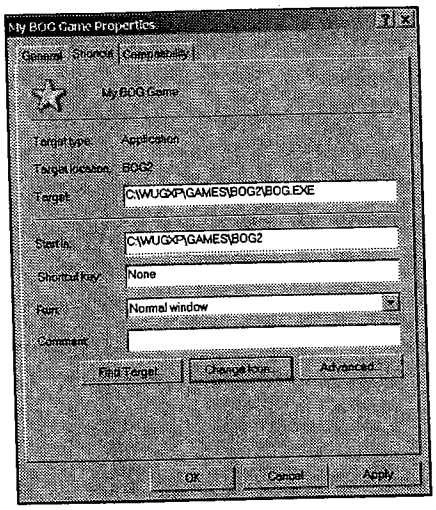
16 Click the **Windows** directory.

17 Click the **system32** directory. Click **shell32.dll.** (If you cannot see these directory names, you may have to go to the Tools menu of My Computer, Folder Options, View, and select Show all files.)



WHAT'S HAPPENING? Windows provides a set of icons in a file named **SHELL32.DLL.** You are going to choose this file by either double clicking it, or selecting it and clicking the open button.

18 Click **Open.** Scroll until you an icon you like. (In this example, a Star was chosen.) Double-click the icon.

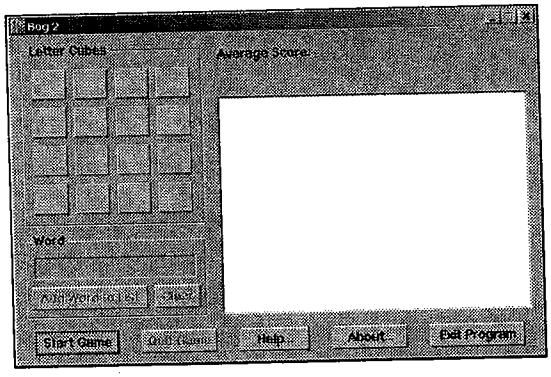


WHAT'S HAPPENING? Now your shortcut to BOG is represented by the icon you selected.

19 Click **Apply**. Click **OK**.

WHAT'S HAPPENING? Your shortcut appears on the desktop with the new icon.

20 Double-click the icon.



WHAT'S HAPPENING? You have used the icon to run the BOG program.

21 Click **Exit Program**.

22 Drag the shortcut to BOG to the **Recycle Bin**.

10.13 Batch Files to Run Windows Programs

The Windows system files will reside in a different directory, depending on how the Windows XP operating system was installed. If the installation was an upgrade to Windows 2000, you will have a WINNT directory. It you upgraded Windows 98 or have a new installation, it will have a Windows directory. The drive may be C:\ or D:\, or another drive. Windows "keeps notes" about itself in the system environ-

ment. Remember that the path was stored in the ***environmental variable*** %Path%. The drive is referred to, in the environment, as "%SystemDrive%" and the directory where the system files are located is referred to as "%SystemRoot%." You can use this information to create a batch file to run the small programs that come with Windows, such as Notepad or Calculator.

Perhaps you prefer to use Notepad when writing batch files, but find it bothersome to have to return to the GUI to start the Notepad program. You can create a batch file that will allow you to run the program without having to return to the desktop. You can also use special features of Notepad to create a log file that will add the current date and time to a file created with Notepad. In order to use this feature, you must create a file with Notepad whose first line is .LOG.

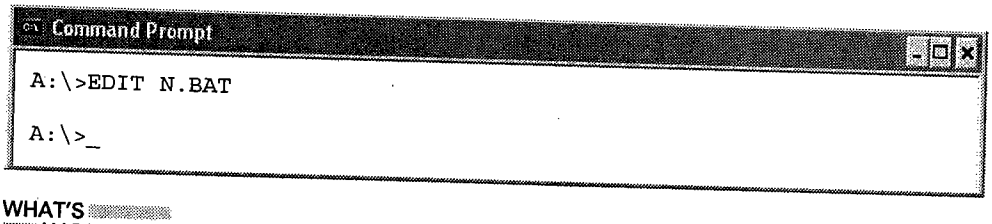
10.14 Activity: Creating a Batch File to Run Notepad

Note: You have shelled out to the Command Line. The DATA disk is in Drive A. A:\> is displayed.

1 Key in the following: `A:\>`**EDIT N.BAT** Enter
%SYSTEMROOT%\NOTEPAD.EXE Enter
A: Enter

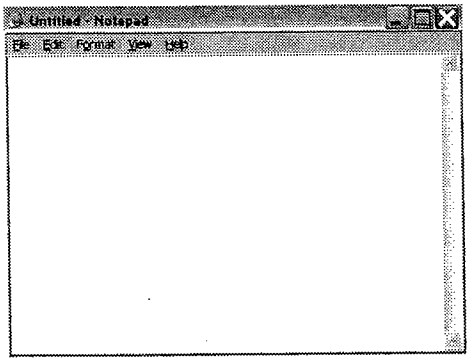
2 Key in the following: Alt + **F**. Press **X**.

3 Key in the following: **Y** Enter



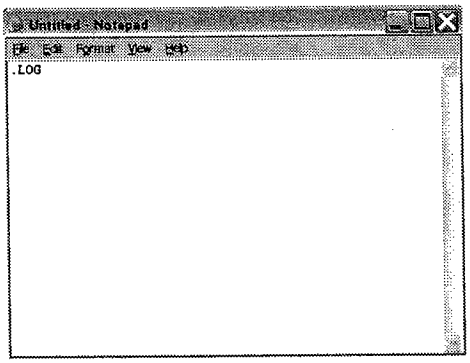
WHAT'S HAPPENING? You have written a batch file to start a Windows applet, Notepad. You used an environmental variable, **%SYSTEMROOT%**. Case does not matter. You could key in an absolute path such as **C:\WINDOWS\NOTEPAD.EXE** or **C:\WINNT\NOTEPAD.EXE** if you knew the name of your Windows directory. However, if you use the environmental variable, Windows knows where the Windows files are located and will substitute the correct name. You did not need to use %SYSTEMDRIVE% as that variable knows which drive the %SYTEMROOT% is on.

4 Key in the following: `A:\>`**N** Enter



WHAT'S HAPPENING? You have opened Notepad without returning to the desktop.

5 Key in the following in the Notepad window: **.LOG** Enter

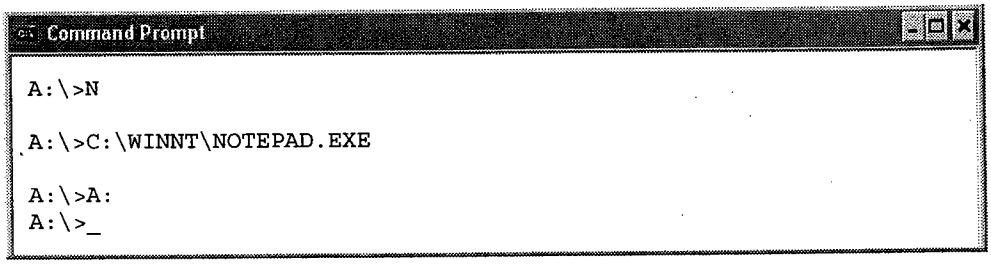


WHAT'S HAPPENING? You are creating a log file using Notepad. Here case *does* matter and you must use uppercase letters preceded by a period.

6 Click **File**. Click **Save As**.

7 In the File name text box, key in the following: **A:\log.txt** Enter

8 Click **Save**. Click **File**. Click **Exit**.

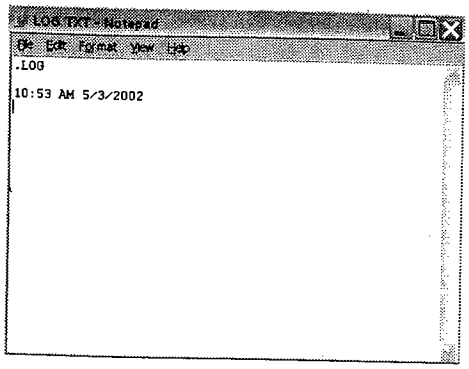


WHAT'S HAPPENING? You have closed Notepad and returned to the Command Line window.

9 Key in the following: **Edit log.bat** Enter
%systemroot%\notepad log.txt Enter
DIR *.AAA Enter

10 Press Alt + **F**. Press **X**. Press **Y**.

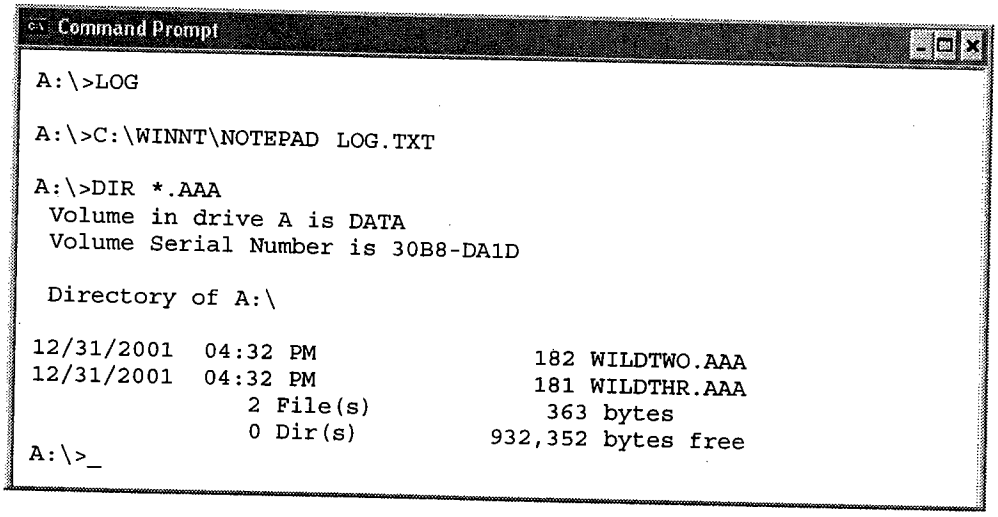
11 Key in the following: `A:\>`**LOG** Enter



WHAT'S HAPPENING? When you supplied a file name with Notepad, it opened the file you specified. As you can see, Notepad has placed the current date and time in the **log.txt** file. However, your second command has not executed. Your batch file requires that you finish using Notepad before it will read the next line and execute it (**DIR *.AAA**).

12 In Notepad, key in the following: **The first entry in my log file.** Enter

13 Press Alt + **F**. Press **X**. Key in the following: **Y**



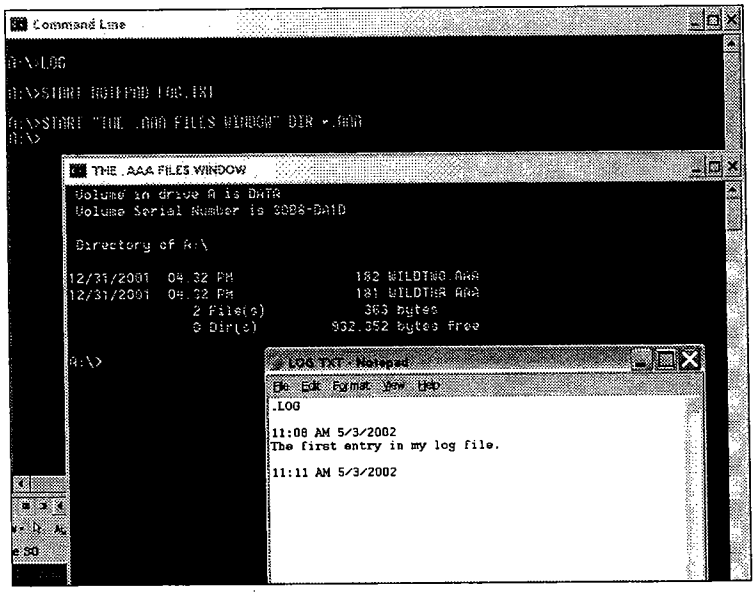
WHAT'S HAPPENING? Now that you exited Notepad, your other command could execute. There is a command called START that allows you to start a program in a new window and at the same time, continue executing your batch file in the previous window. You may also change the title of the Command Line window.

14 Edit and save the LOG.BAT file so it reads as follows and only has these two lines:

START notepad log.txt
START "THE .AAA FILES WINDOW" DIR *.AAA

WHAT'S HAPPENING? The START command will start a new command window so that LOG.TXT will open in one window. The second command accomplishes two tasks. It will open another window, and it will give the window the title enclosed in quotation marks.

15 Key in the following: **LOG** Enter



WHAT'S HAPPENING? You have three windows open, your original Command Line, your Notepad window, and "THE .AAA FILES WINDOW." The Notepad window is waiting for you to make an entry.

16 Make the Notepad window active. In the Notepad window, key in the following: **The second entry in my log file.**

17 Click **File**. Click **Save**. Click **File**. Click **Exit**.

WHAT'S HAPPENING? You still have THE AAA FILES WINDOW open. The DIR command executed.

18 Key in the following in each of the command line windows: **Exit** Enter

WHAT'S HAPPENING? You have closed the Command Line windows.

10.15 Special Batch File Commands

There are commands specifically designed to be used in batch files. They are the same for Windows XP as they were for Windows 2000. These commands can make batch files extremely versatile. They are listed in Table 10.2 below:

Command	Purpose
CALL	Calls one batch program from another without causing the first batch program to stop. The CALL command now accepts labels as the target of the call.
ECHO	Displays or hides the text in batch programs while the program is running. Also used to determine whether or not commands will be *echoed* to the screen while the program file is running.

ENDLOCAL	Ends localization of environment changes in a batch file, restoring environment variables to their values before the matching SETLOCAL command.
FOR	Runs a specified command for each file in a set of files. This command can also be used at the command line.
GOTO	Directs the operating system to a new line in the program that you specify with a label.
IF	Performs conditional processing in a batch program, based on whether or not a specified condition is true or false.
PAUSE	Suspends processing of a batch file and displays a message prompting the user to press a key to continue.
REM	Used to document your batch files. The operating system ignores any line that begins with REM, allowing you to place lines of information in your batch program or to prevent a line from running.
SETLOCAL	Begins localization of environmental variables in a batch file. Localization lasts until a matching ENDLOCAL command is encountered or the end of the batch file is reached.
SHIFT	Changes the position of the replaceable parameter in a batch program.

Table 10.2—Batch File Commands

You will examine and use some of these commands in the following activities.

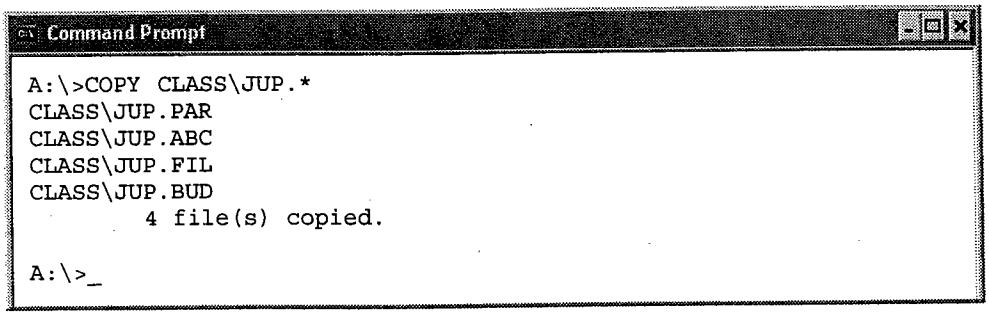
10.16 The REM Command

The REM command, which stands for "remarks," is a special command that allows the user to key in explanatory text that will be displayed on the screen. Nothing else happens. REM does not cause the operating system to take any action, but it is very useful. When a line begins with REM, the operating system knows that anything following the REM is not a command and, thus, is not supposed to be executed, just displayed on the screen. REM allows a batch file to be ***documented***. In a data-processing environment, "to document" means to give an explanation about the purpose of a program. This process can be very important when there are many batch files on a disk, especially when someone who did not write the batch file would like to use it. The REM statements should tell anyone what the purpose of the batch file is. The remarks can also include the name of the batch file, the time and date it was last updated, and the author of the batch file.

10.17 Activity: Using REM

Note: You have shelled out to a Command Line window. The DATA disk is in Drive A. A:\> is displayed.

1 Key in the following: `A:\>`**COPY CLASS\JUP.*** Enter



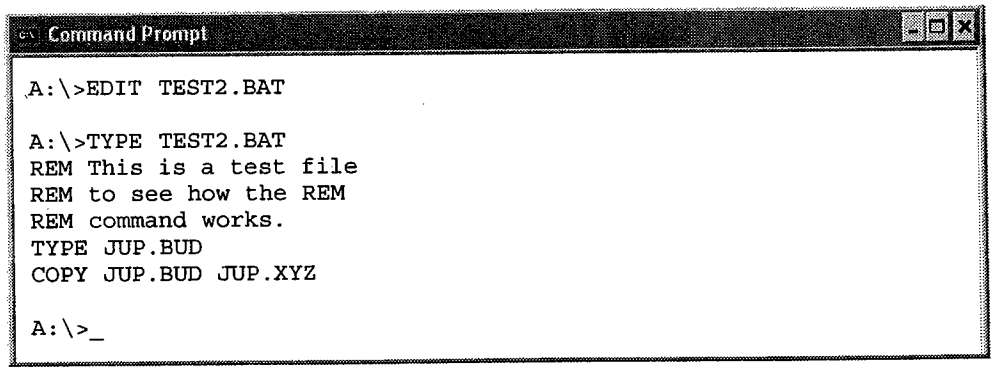
WHAT'S HAPPENING? You have copied some files that were previously moved to the CLASS directory to the root of the DATA disk.

2 Key in the following: `A:\>`**EDIT TEST2.BAT** Enter
REM This is a test file Enter
REM to see how the REM Enter
REM command works. Enter
TYPE JUP.BUD Enter
COPY JUP.BUD JUP.XYZ

WHAT'S HAPPENING? You are using Edit to write another batch file called **TEST2.BAT**. You have inserted some text with REM preceding each line. You keyed in two command line commands, TYPE and COPY. Now you want to save this file to the disk and return to the system level.

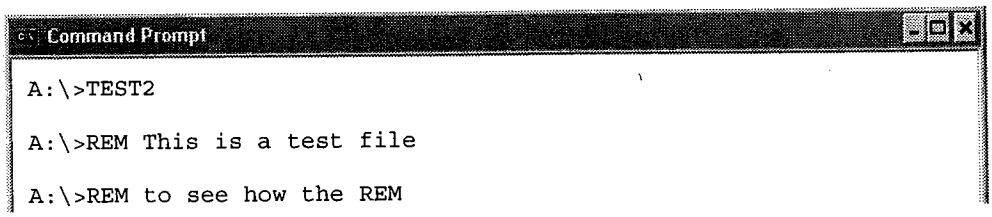
3 Press Alt + **F**. Press **X**. Press **Y**.

4 Key in the following: `A:\>`**TYPE TEST2.BAT** Enter



WHAT'S HAPPENING? This batch file was created as a test case. The remarks just keyed in explain the purpose of this batch file. You created **TEST2.BAT** in Edit and returned to the system prompt. You then displayed **TEST2.BAT** with the TYPE command. To execute the **TEST2.BAT** batch file, you must run it.

5 Key in the following: `A:\>`**TEST2** Enter



```
A:\>REM command works.

A:\>TYPE JUP.BUD

Jupiter is the largest planet
in our Solar System.  It has a
giant red spot on it.  Huge storms
larger than our earth that last
more than a century take place
on the planet Jupiter.

A:\>COPY JUP.BUD JUP.XYZ
        1 file(s) copied.

A:\>_
```

WHAT'S HAPPENING? When you keyed in **TEST2**, the batch file was executed. The operating system read the first line of the batch file, *REM This is a test file.* It knew that it was supposed to do nothing but display the text following REM on the screen. Then the next line in the batch file was read, *REM to see how the REM*, and the same procedure was followed. The operating system kept reading and displaying the REM lines until it got to the line that had the command TYPE. To the operating system, TYPE is a command, so it executed or ran the TYPE command with the parameter JUP.BUD. Then the next line was read, which was another command, COPY, so it was executed. The file **JUP.BUD** was copied to a new file called **JUP.XYZ**. Then the operating system looked for another line in the batch file but could find no more lines, so it returned to the system level. The purpose of REM is to provide explanatory remarks about the batch file.

10.18 The ECHO Command

Notice in the above activity, when you ran TEST2.BAT you saw the command on the screen, and then the command executed. You saw the words "TYPE JUP.BUD" which was the command, and then saw the typed-out file, the results of the command. Both the command and the output of the command were "echoed" to the screen. ECHO is a command that means display to the screen. The default value for ECHO is on. Unless specifically told otherwise, both commands and their results will show on the screen. In a batch file, you can turn off the display of the command and see only the output of a command—not the command itself. For instance, COPY THIS.FIL THAT.FIL is a command. The output of the command is **1 File(s) copied**. The work of the command is the actual copying of the file. See Table 10.3.

	Echo On Display	Echo Off Display
Command:	COPY THIS.FIL THAT.FIL	
Output:	1 File(s) copied	1 File(s) copied

Table 10.3—ECHO On or Off

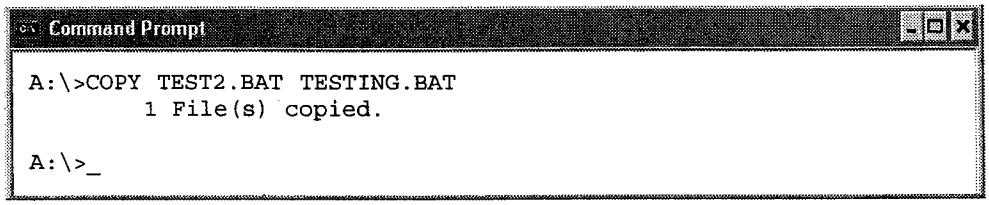
If the purpose of the REM command is to document a batch file, what is the purpose of the ECHO command? One purpose of the ECHO command is, by turning it off, you can minimize screen clutter. For instance, although you want to

use the REM command to document your batch file, you really only want to see this documentation when you type out the contents of the file, or edit it to make changes. You do not need to see your documentation on the screen every time you run the batch file. ECHO OFF allows you to suppress the display of the commands.

10.19 Activity: Using ECHO

Note: The DATA disk is in Drive A. A:\> is displayed.

1 Key in the following: A:\>**COPY TEST2.BAT TESTING.BAT** Enter



WHAT'S HAPPENING? You made a copy of the file **TEST2.BAT**.

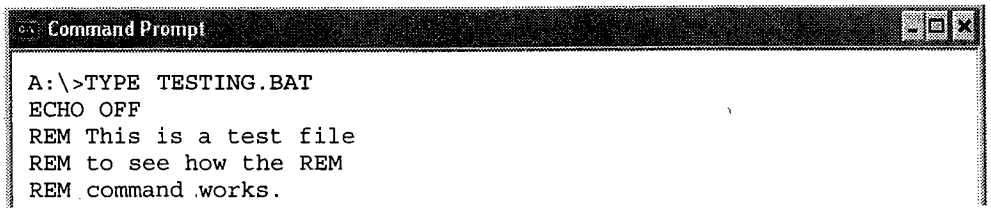
2 Key in the following: A:\>**EDIT TESTING.BAT** Enter

3 At the top of the file, key in the following: **ECHO OFF** Enter

WHAT'S HAPPENING? You are using a copy of the batch file from the previous activity. The only difference is that you added one line at the top of the batch file to turn ECHO off. You are going to run the batch file so that only the output of each command is displayed, not the actual commands. First you must exit Edit and save the file to the disk.

4 Press Alt + **F**. Press **X**. Press **Y**.

5 Key in the following: A:\>**TYPE TESTING.BAT** Enter



```
TYPE JUP.BUD
COPY JUP.BUD JUP.XYZ

A:\>_
```

WHAT'S HAPPENING? You saved the file as **TESTING.BAT** and displayed the contents on the screen. Now you wish to execute the file.

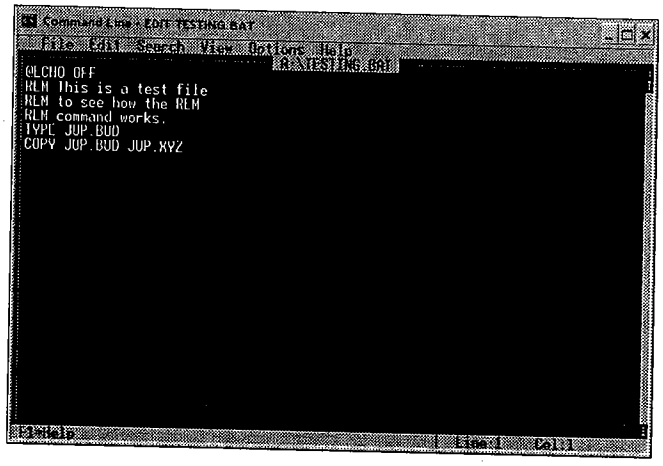
6 Key in the following: `A:\>`**TESTING** Enter

```
Command Prompt
A:\>TESTING

A:\>ECHO OFF

Jupiter is the largest planet
in our Solar System.  It has a
giant red spot on it.  Huge storms
larger than our earth that last
more than a century take place
on the planet Jupiter.
        1 file(s) copied.

A:\>_
```

WHAT'S HAPPENING? The batch file **TESTING.BAT** has the same commands as **TEST2.BAT**, but this time you saw only the output of the commands, not the actual commands themselves. You saw the ECHO OFF command on the screen, but you did not see the REM commands displayed on the screen. You saw the results of the TYPE JUP.BUD command, the contents of the file on the screen, but you never saw the TYPE JUP.BUD command on the screen. You also did not see the COPY JUP.BUD JUP.XYZ command, only the results of the command—the message "1 file(s) copied.

You can eliminate the display of the ECHO OFF command.

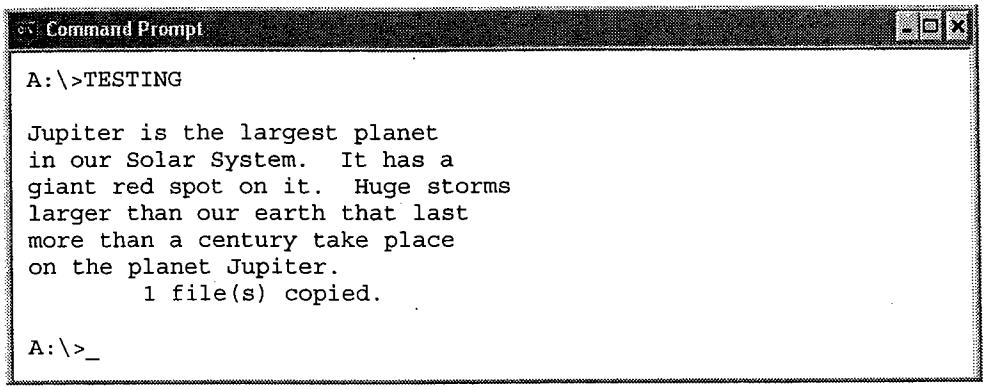
7 Key in the following: `A:\>`**EDIT TESTING.BAT** Enter
@ (do not press Enter)



WHAT'S HAPPENING? You have inserted the @ symbol in front of ECHO OFF to prevent the command from echoing to the screen.

8 Press **Alt** + **F**. Press **X**. Press **Y**.

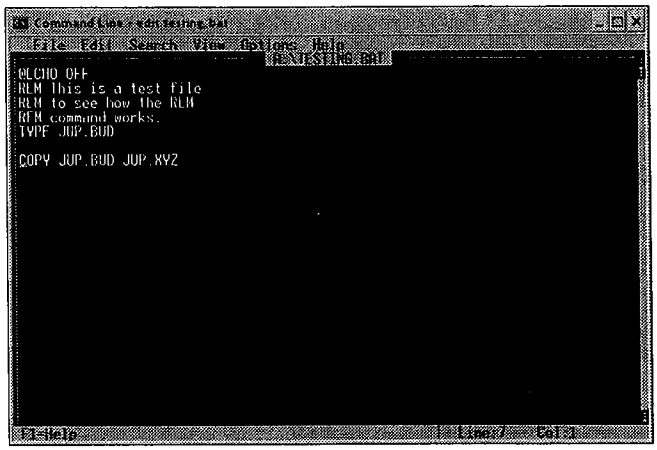
9 Key in the following: `A:\>`**TESTING** **Enter**



WHAT'S HAPPENING? The ECHO OFF display was suppressed. All you see at the end of the display is 1 file(s) copied. This may be misleading. You want to see the names of the source and destination files used in the copy process.

10 Key in the following: **EDIT TESTING.BAT** **Enter**

11 Place the cursor under the C in COPY and press the **Enter** key.

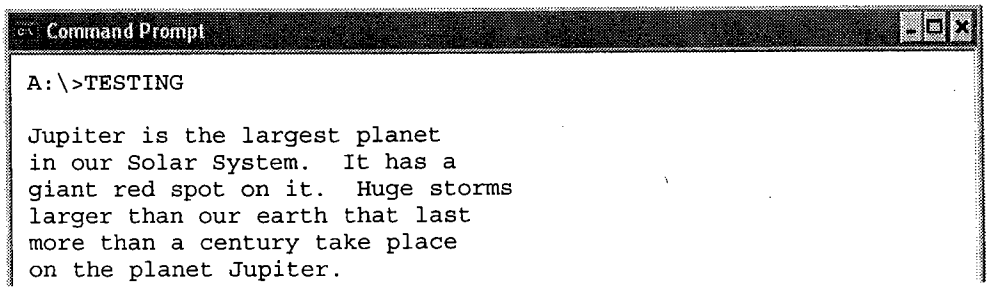


WHAT'S HAPPENING? There is now a blank line in the file.

12 Place the cursor in the blank line and Key in the following: **ECHO ON**

13 Press **Alt** + **F**. Press **X**. Press **Y**.

14 Key in the following: **TESTING** **Enter**



```
A:\>COPY JUP.BUD JUP.XYZ
        1 file(s) copied.

A:\>_
```

WHAT'S HAPPENING? You have manipulated the screen display to better suit your needs using ECHO OFF and ECHO ON.

You already have a file by the name of **JUP.XYZ**, but, even though you are using the COPY command, it did not tell you that the file already exists (overwrite protection). The purpose of using a batch program would be defeated if there were interaction required by the user, so the warning is not there. The differences between ECHO ON and ECHO OFF are exemplified in Table 10.4.

	TEST2.BAT— ECHO ON Display	TESTING.BAT— ECHO OFF Display
Command:	ECHO ON	ECHO OFF
Command:	REM This is a test file	
Command:	REM to see how the REM	
Command:	REM command works.	
Command:	TYPE JUP.BUD	
Output:	Jupiter is the largest planet in our Solar System. It has a giant red spot on it. Huge storms larger than our earth that last more than a century take place on the planet Jupiter.	Jupiter is the largest planet in our Solar System. It has a giant red spot on it. Huge storms larger than our earth that last more than a century take place on the planet Jupiter.
Command:	COPY JUP.BUD JUP.XYZ	
Output:	1 File(s) copied	1 File(s) copied

(*Note:* Although commands and file names are shown as uppercase letters, the case does not matter.)

Table 10.4—ECHO ON and ECHO OFF: A Comparison of Screen Displays

10.20 The CLS Command

In the previous exercise, running the batch file TESTING caused the output of that file to be placed on the screen directly below the request (TESTING) for the file to execute. In many instances, the screen already contained previously executed commands and outputs. The purpose of this batch file is to 1) display the contents of a file and 2) to copy that file to another file. Having the display appear on a screen that already contains information can be difficult to read. Using the CLS (CLear Screen) command after turning the ECHO off eliminates this problem.

10.21 Activity: Using CLS

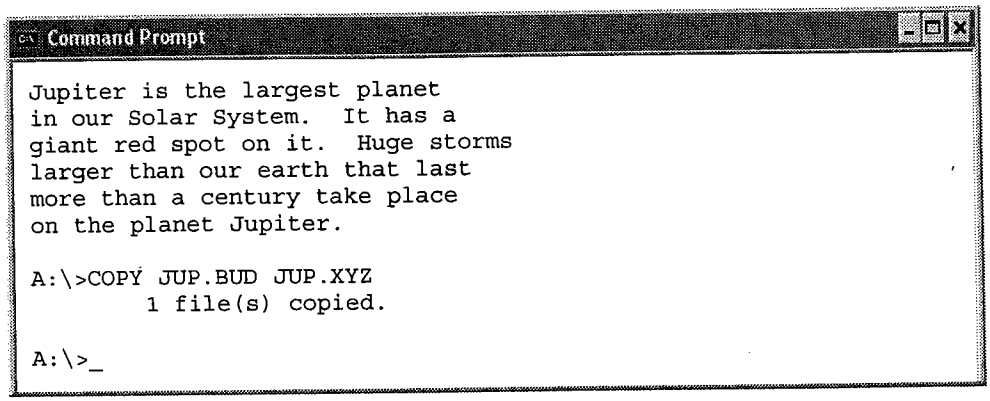
1 Key in the following: **EDIT TESTING.BAT** Enter

2 Place the cursor under the R in the first REM command and press the Enter key.

3 In the blank line created, key in the following: **CLS**

4 Press Alt + **F**. Press **X**. Press **Y**.

5 Key in the following: **TESTING** Enter



WHAT'S HAPPENING? The first "action" taken by the batch file after the ECHO was turned off was to clear the screen. When and if to turn the ECHO off and on, and when and if to clear the screen depends on the purpose and function of each batch file.

10.22 The PAUSE Command

Another batch file command is PAUSE, which does *exactly* what its name implies: It tells the batch file to stop executing until the user takes some action. Batch file processing is suspended, and no other batch command will be executed until the user presses a key. The PAUSE command will wait forever until the user takes some action.

10.23 Activity: Using PAUSE

Note: The DATA disk is in Drive A. A:\> is displayed.

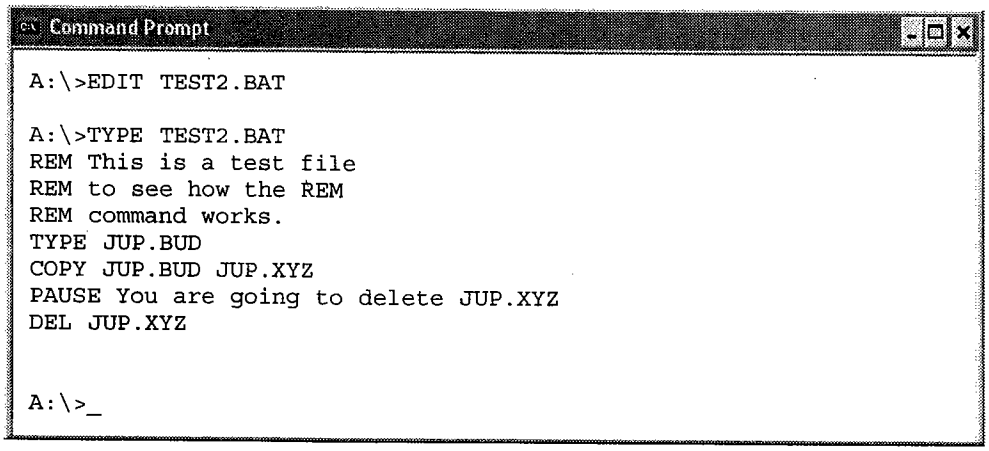
1 Key in the following: A:\>**EDIT TEST2.BAT** Enter

2 Press Ctrl + End

3 Key in the following: **PAUSE You are going to delete JUP.XYZ** Enter
DEL JUP.XYZ

4 Press Alt + **F**. Press **X**. Press **Y**.

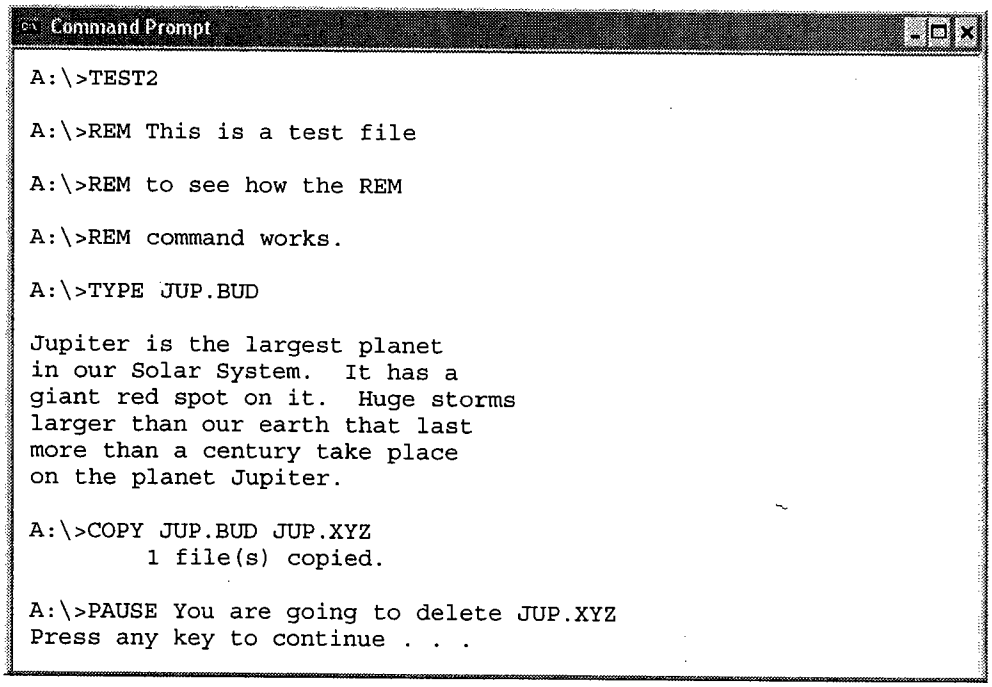
5 Key in the following: A:\>**TYPE TEST2.BAT** Enter



WHAT'S HAPPENING? You saved the file to disk with the changes you made. You then looked at the contents of the file with the TYPE command. You edited the batch file **TEST2.BAT**. When the file is executed, the first three lines of the file, the **REM** statements, explain the purpose of **TEST2.BAT**. Then the batch file displays the contents of **JUP.BUD** on the screen and copies the file **JUP.BUD** to a new file, **JUP.XYZ**. The PAUSE statement tells you that the file is going to be deleted and gives you a chance to change your mind. After you take action by pressing a key, the file **JUP.XYZ** is erased.

To execute **TEST2.BAT**, you must key in **TEST2** at the prompt.

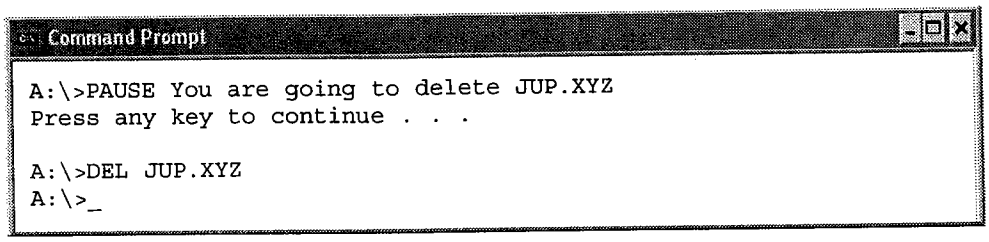
6 Key in the following: `A:\>`**TEST2** Enter



WHAT'S HAPPENING? The batch file TEST2 has stopped running or "paused." It has halted execution until some action is taken. When you press a key, the operating system will read and execute the next line of the batch file. PAUSE just stops; it is

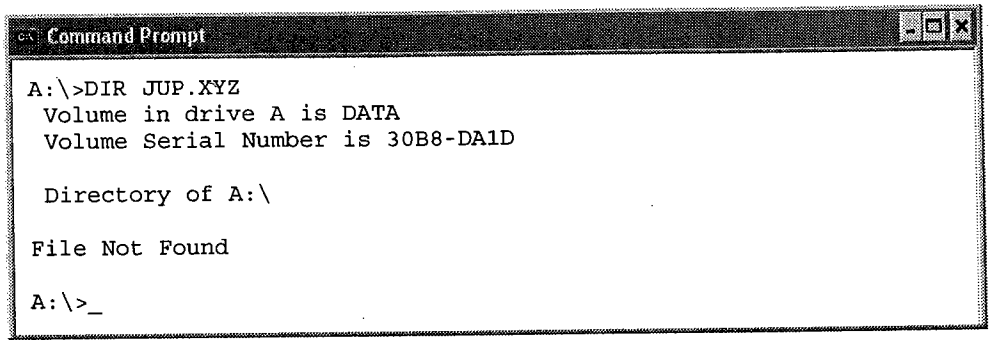
not an order. If ECHO were off, all you would see is the message, "Press any key to continue ...". You would not see the message, "You are going to delete JUP.XYZ."

7 Press Enter



WHAT'S HAPPENING? The batch file continued executing all the steps and deleted the file called **JUP.XYZ**.

8 Key in the following: `A:\>`**DIR JUP.XYZ** Enter



WHAT'S HAPPENING? The file **JUP.XYZ** was deleted.

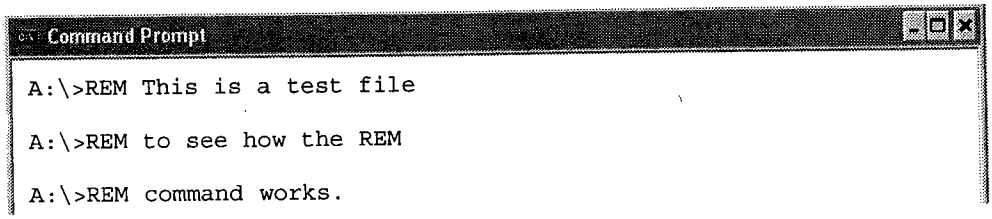
10.24 Stopping a Batch File from Executing

In the above activity, you pressed a key after the PAUSE command was displayed so that the batch file continued to execute. What if you wanted to stop running the batch file? You can do this by interrupting or exiting from a running batch file. You do this by pressing the Ctrl key, and while pressing the Ctrl key, pressing the letter C (Ctrl + C or Ctrl + Break). At whatever point Ctrl + C is pressed, you leave the batch file and return to the system prompt. The rest of the lines in the batch file do not execute.

10.25 Activity: Quitting a Batch File

Note: The DATA disk is in Drive A. A:\> is displayed.

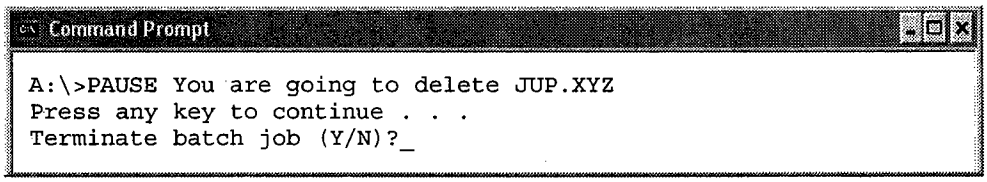
1 Key in the following: `A:\>`**TEST2** Enter



```
A:\>TYPE JUP.BUD

Jupiter is the largest planet
in our Solar System.  It has a
giant red spot on it.  Huge storms
larger than our earth that last
more than a century take place
on the planet Jupiter.

A:\>COPY JUP.BUD JUP.XYZ
        1 file(s) copied.

A:\>PAUSE You are going to delete JUP.XYZ
Press any key to continue . . .
```

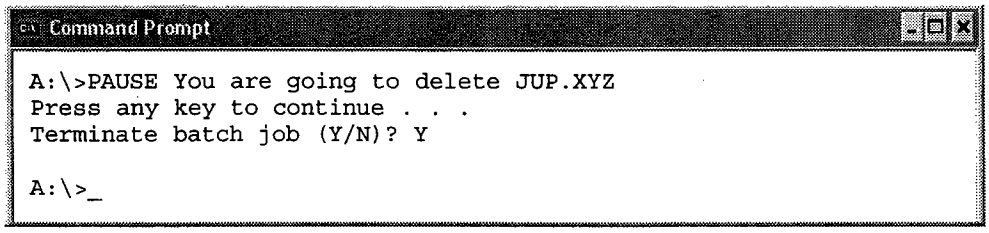
WHAT'S HAPPENING? You are at the same point as you were in the last activity. The batch file reached the PAUSE command. It has momentarily stopped running. You do not want to erase **JUP.XYZ**. You want the batch file to cease operation. Previous experience with the PAUSE command showed that pressing any key would continue running the program. If any key were pressed here, the next line in the file, **DEL JUP.XYZ**, would execute and the file **JUP.XYZ** would be erased. To stop this from happening, another action must be taken to interrupt the batch file process.

2 Hold down the Ctrl key, and while it is down, press the letter **C**. Then release both keys.



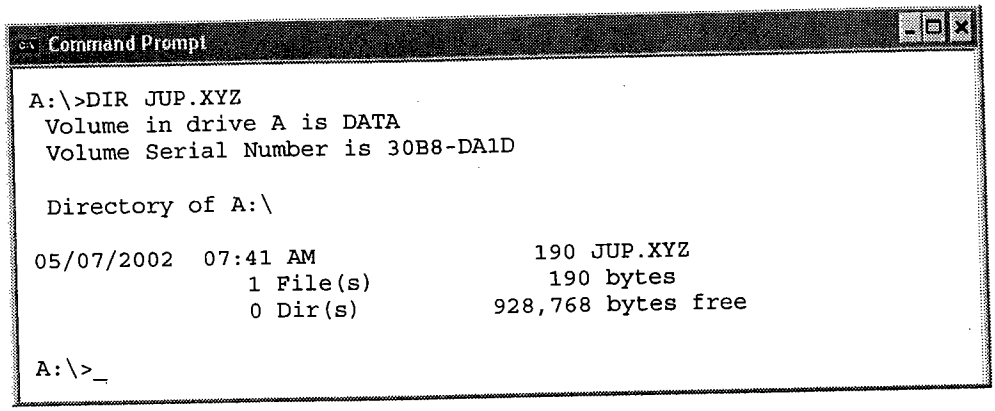
WHAT'S HAPPENING? The message is giving you a choice: either stop the batch file from running (**Y** for "yes") or continue with the batch file (**N** for "no"). If you press **Y**, the last line in the batch file, DEL JUP.XYZ, will not execute.

3 Press **Y** Enter



WHAT'S HAPPENING? The system prompt is displayed. If the batch file was interrupted properly, **JUP.XYZ** should not have been deleted because the line, DEL JUP.XYZ should not have executed.

4 Key in the following: `A:\>`**DIR JUP.XYZ** Enter



WHAT'S HAPPENING? The file **JUP.XYZ** is still on the DATA disk. Pressing Ctrl + C at the line *PAUSE You are going to delete JUP.XYZ* broke into the batch file **TEST2.BAT** and stopped it from running. Because **TEST2.BAT** stopped executing and returned you to the system prompt, it never got to the command line DEL JUP.XYZ. Therefore, the file **JUP.XYZ** is still on the DATA disk. Although in this activity you broke into the batch file at the PAUSE statement, you can press Ctrl + C any time during the execution of a batch file. The batch file will stop when it has completed the current command before executing the next one. The problem is, with the speed of today's computers, it is difficult to ascertain how many lines of the batch file have been read by the operating system when you press Ctrl + C.

10.26 Replaceable Parameters in Batch Files

In the same way that you use parameters with system commands, you can use parameters effectively in batch files. For instance, look at the command DIR A: /W.

```
Command      Command Line Parameter
DIR          A: /W
```

In the above example, the space and the / are delimiters. DIR is the command. A: and W are parameters that tell the operating system that you want a directory of A: and that you want it displayed in a wide mode. Parameters give the command additional instructions on what to do. When you use the DIR command as used above, the /W parameter is fixed; you cannot choose another letter to accomplish a wide mode display.

Many commands use ***variable*** or ***replaceable parameters***. An example of a command that uses a replaceable parameter is TYPE. TYPE requires one parameter, a file name, but the file name you use will vary; hence, it is a variable parameter. The TYPE command uses the parameter that you keyed in to choose the file to display on the screen. You can key in TYPE THIS.FIL or TYPE TEST.TXT or whatever file name you want. You replace the file name for the parameter, hence the term replaceable parameter.

```
Command          Replaceable Command Line Parameter
 TYPE                    THIS. FIL
```

or

```
Command              Replaceable Command Line Parameter
 TYPE                        TEST.TXT
```

Batch files can also use replaceable parameters, also called ***dummy parameters***, ***substitute parameters***, or ***positional parameters***. When you write the batch file, you insert ***place holders*** that will hold information that is keyed in at the time the batch file is executed. When you execute the batch file by keying in the batch file name, you also key in additional information on the command line that will be inserted where the placeholders are in the batch file. The batch file looks to the command line and selects which information you desire by its position (positional parameters) on the command line. What you are doing is parsing a command. To *parse* is to analyze something in an orderly way. In linguistics, to parse is to divide words and phrases into different parts in order to understand relationships and meanings. In computers, to parse is to divide the computer language statement into parts that can be made useful for the computer. In the above example, the TYPE command had the argument TEST.TXT passed to it so that it can display the contents of that variable. When you write the batch file, you supply the place holder or marker to let the batch file know that something, a variable, will be keyed in with the batch file name at execution time. The place holder, marker, or blank parameter used in a batch file is the percent sign (%) followed by a number from **0** through **9**. The **%** sign is the signal to the operating system that a parameter is coming. The numbers indicate what position the parameter is on the command line. Whatever is first is **%0**, usually the command itself. Thus, the command occupies the position of %0. The next item on the command line is in the first position (%1) and the next item on the command line is in the second position (%2) and so on.

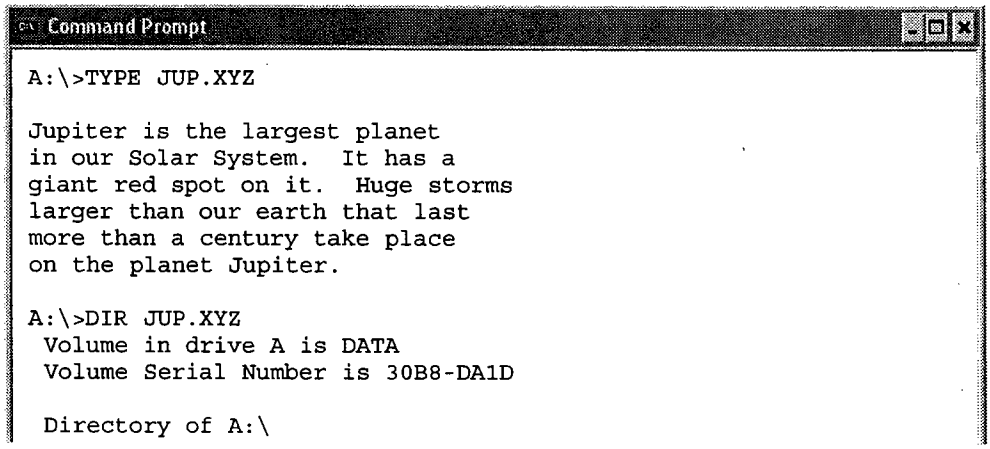
The batch files that you have written so far deal with specific commands and specific file names, but the real power of batch files is their ability to use replaceable parameters. You are going to write a batch file in the usual way with specific file names, and then use the batch file to see how replaceable parameters work.

10.27 Activity: Using Replaceable Parameters

Note: The DATA disk is in Drive A. A:\> is displayed.

1 Key in the following: `A:\>`**TYPE JUP.XYZ** Enter

2 Key in the following: `A:\>`**DIR JUP.XYZ** Enter



```
05/07/2002  07:41 AM               190 JUP.XYZ
               1 File(s)            190 bytes
               0 Dir(s)         928,768 bytes free

A:\>_
```

WHAT'S HAPPENING? This file was created in the last activity. It has data in it and occupies 190 bytes of space on the disk. (Your file size may differ slightly.) If you remember, when you delete a file, the data is still on the disk. What if you wanted a way to delete the data completely so that it cannot ever be recovered? You can overwrite the file with new data. If you key in ECHO at the command line, you get a status report of whether ECHO is on or off. You can redirect the output of the ECHO command to your file, making it 13 bytes long and replacing the data in the file with the output of the ECHO command. You are going to first try this at the command line.

3 Key in the following: `A:\>`**ECHO > JUP.XYZ** Enter

4 Key in the following: `A:\>`**TYPE JUP.XYZ** Enter

5 Key in the following: `A:\>`**DIR JUP.XYZ** Enter

Command Prompt

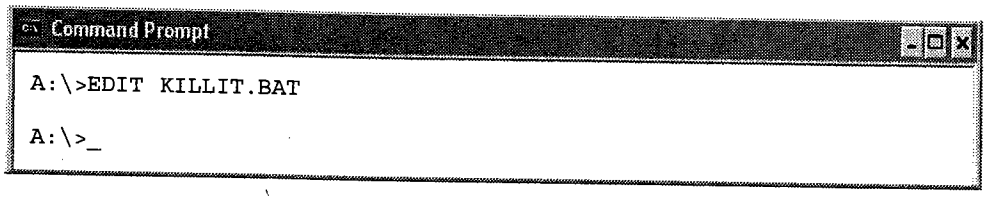
```
A:\>ECHO > JUP.XYZ

A:\>TYPE JUP.XYZ
ECHO is on.

A:\>DIR JUP.XYZ
 Volume in drive A is DATA
 Volume Serial Number is 30B8-DA1D

 Directory of A:\

05/03/2002  02:26 PM                13 JUP.XYZ
               1 File(s)             13 bytes
               0 Dir(s)         928,768 bytes free

A:\>_
```

WHAT'S HAPPENING? As you can see, it worked. You have *really* overwritten this file. Your data is no longer in the file to recover. This command would be useful when deleting files of a confidential nature. It would prevent most data recovery programs from being able to recover the data from your file. It can also be used in a batch file.

6 Key in the following: `A:\>`**EDIT KILLIT.BAT** Enter
ECHO > JUP.XYZ Enter
TYPE JUP.XYZ Enter
DEL JUP.XYZ

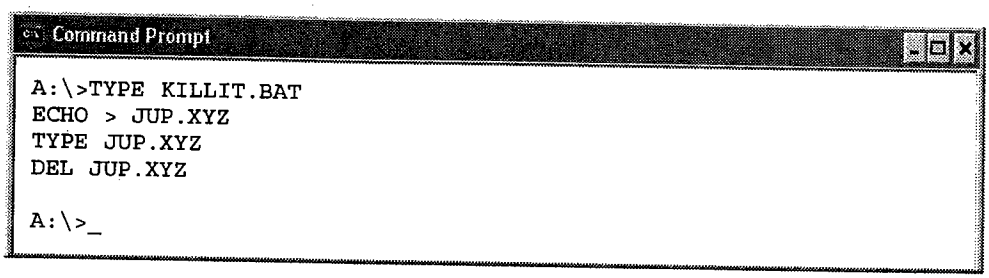
7 Press Alt + **F**. Press **X**.

8 Press **Y**.



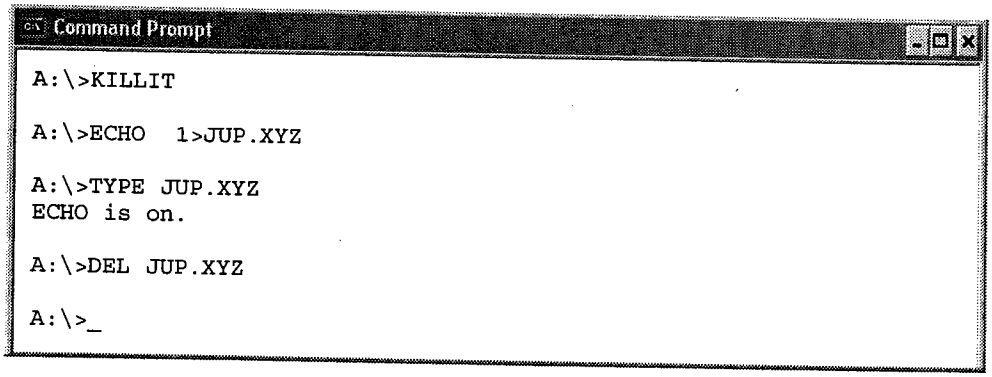
WHAT'S HAPPENING? Edit is the tool you used to write the batch file. You created a simple batch file that sends data to **JUP.XYZ**, displays the file contents, and then deletes the file called **JUP.XYZ**. You then used the Edit menu to exit and save **KILLIT.BAT** to your disk.

9 Key in the following: `A:\>`**TYPE KILLIT.BAT** Enter



WHAT'S HAPPENING? You displayed the contents of the **KILLIT.BAT** file. To execute this batch file, you must *call* it, which is another way of saying key in the command name—the name of the batch file.

10 Key in the following: `A:\>`**KILLIT** Enter



WHAT'S HAPPENING? The batch file called **KILLIT** ran successfully. (The 1 before the directional sign (>) indicates the numerical value of Standard Output, indicating that the Standard Ouput of the Echo command is redirected to the object of the directional sign.) However, this batch file is not useful, as it can be used only for the file called **JUP.XYZ**. You have deleted **JUP.XYZ** so now **KILLIT.BAT** is no longer useful.

What if you wanted to do the same sequence of commands for a file called **JUP.TMP** or **PERSONAL.FIL** or any other file on the disk? Until now, you would have to create another batch file using **JUP.TMP** instead of **JUP.XYZ**. You would write another batch file for **PERSONAL.FIL**. You can quickly clutter up your disks with many batch files, all doing the same thing but using different file names and having no value after they have executed. An easier way is to have a batch file that does the same steps—a generic batch file. When you execute it, you supply the specific parameter or file name that interests you. When you write this batch file, you need to supply a place for the name of the file. These places are called "replaceable parameters." They are percent signs followed by numbers.

You are going to edit **KILLIT.BAT** so that it uses replaceable parameters. In addition, you will document it and add some protection for yourself. When you key in the replaceable parameters, be sure to use the percent sign (%), then the number **1**, and not the lowercase of the letter L (**l**). Also note that there is no space between % and the number **1**.

11 Key in the following:

`A:\>`**EDIT KILLIT.BAT** Enter
REM This batch file will make Enter
REM the data in a file difficult to recover. Enter
DIR %1 Enter
PAUSE You are going to kill the file, %1. Are you sure? Enter

12 Replace **ECHO > JUP.XYZ** with **ECHO > %1**.

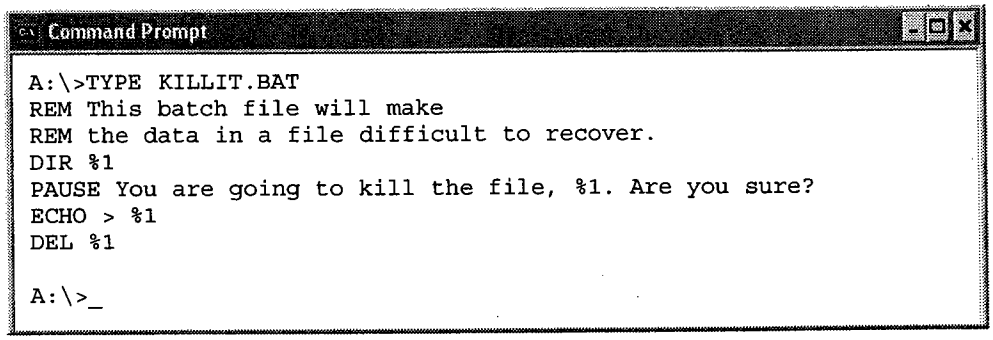
13 Delete the line **TYPE JUP.XYZ**.

14 Replace **DEL JUP.XYZ** with **DEL %1**.

15 Press Alt + **F**. Press **X**.

16 Press **Y**.

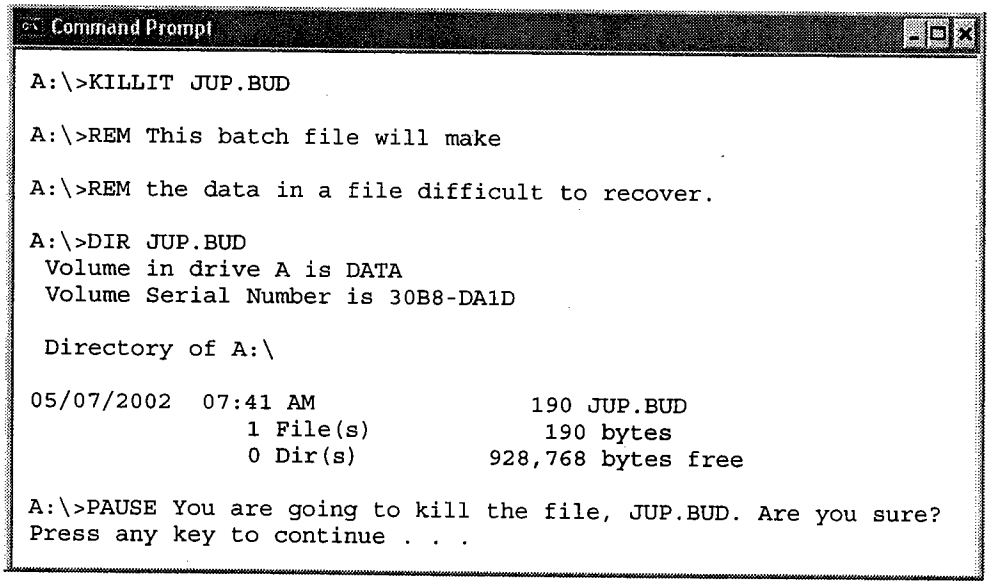
17 Key in the following: `A:\>`**TYPE KILLIT.BAT** Enter



WHAT'S HAPPENING? You used Edit to edit the file **KILLIT.BAT**. You then saved it to the disk. You displayed the contents of the file on the screen. The contents of the batch file **KILLIT.BAT** are different from the previous version of **KILLIT.BAT**. By using the place holder **%1**, instead of a specific file name, you are saying that you do not yet know what file name (**%1**) you want these commands to apply to. When you run the batch file **KILLIT**, you will provide a value or parameter on the command line that the batch file will substitute for **%1**. For instance, if you key in on the command line, **KILLIT MY.FIL**, **KILLIT** is in the zero position on the command line (**%0**) and **MY.FIL** is in the first position on the command line (**%1**). Remember, you used %1 as a place holder in the KILLIT.BAT file. You will fill that place holder at the time of execution.

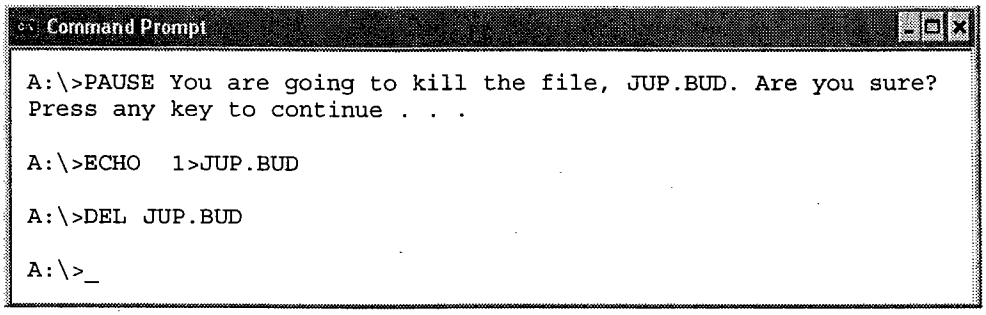
For you to understand the purpose of replaceable parameters, it is helpful to view them as *positional* parameters. The operating system gets the information or knows what to substitute by the position on the command line. The first piece of data on the command line is always in position 0; the second piece of data on the command line is always in position 1; the third piece of data on the command line is always in position 2, and so on.

18 Key in the following: `A:\>`**KILLIT JUP.BUD** Enter



WHAT'S HAPPENING? In the command line **KILLIT JUP.BUD**, **KILLIT** is position 0 and **JUP.BUD** is position 1. The batch file **KILLIT** executed each command line. However, when it found **%1** in the batch file, it looked for the first position after **KILLIT** on the command line, which was **JUP.BUD**. It substituted **JUP.BUD** every time it found **%1**. You placed the **DIR %1** in the batch file to confirm that it is on the disk. The PAUSE statement allows you to change your mind. The **%1** in the PAUSE line identifies which file is to be killed.

19 Press Enter



WHAT'S HAPPENING? You have deleted the file. Even if you, or anyone else, try to recover it, the data is truly gone. You have written a generic or "plain wrap" batch file that allows you to use the same batch file over and over. All you have to supply is a value or parameter after the batch file name on the command line. Thus, you could key in **KILLIT BUSINESS.APP, KILLIT SALES.LET, KILLIT FEB.99, KILLIT TELE.SET**, or any other file name. The batch file will execute the same commands over and over, using the position 1 (%1) value (the file name) you key in after the batch file name. You can see that because this file is versatile, it is infinitely more useful than it was without positional parameters.

10.28 Multiple Replaceable Parameters in Batch Files

In the above example, you used one replaceable parameter. What happens if you need more than one parameter? For instance, if you want to include the COPY command in a batch file, COPY needs two parameters: *source* and *destination*. Many commands require more than one parameter. You may also use multiple parameters in batch files. You can have up to 10 dummy parameters (%0 through %9). Remember, replaceable parameters are sometimes called positional parameters because the operating system uses the position number in the command line to determine which parameter to use. The parameters are placed in order from left to right. For example, examine the command line:

COPY MYFILE.TXT YOUR.FIL

COPY is in the first position, %0 (computers always count beginning with 0, not 1). MYFILE.TXT is in the second position, %1, and YOUR.FIL is in the third position, %2.

The next activity will allow you to create a simple batch file with multiple replaceable parameters so you will see how the positional process works. Then you will write another batch file, and in it you will create a command that the operating system does not have. Your new command will copy all files *except* the ones you specify.

10.29 Activity: Using Multiple Replaceable Parameters

Note: The DATA disk is in Drive A. A:\> is displayed.

1 Key in the following:

A:\>**EDIT MULTI.BAT** Enter

REM This is a sample batch file Enter

REM using more than one replaceable parameter. Enter

TYPE %3 Enter

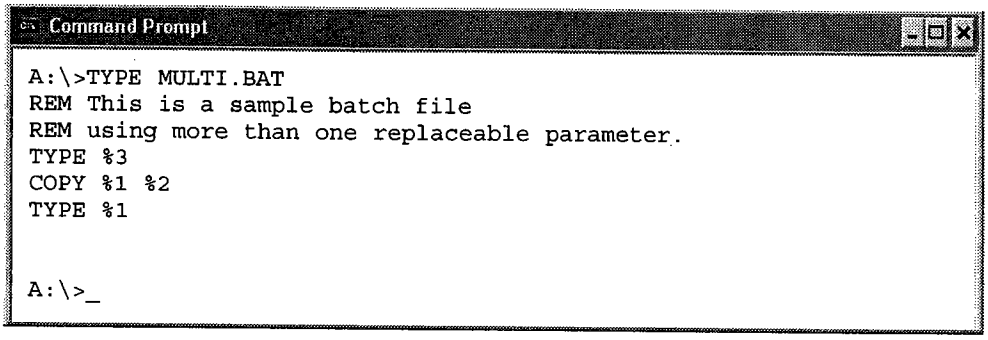
COPY %1 %2 Enter

TYPE %1

2 Press Alt + **F**. Press **X**.

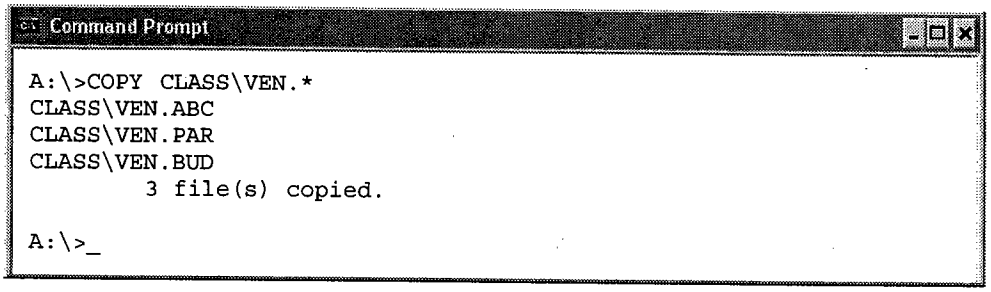
3 Press **Y**.

4 Key in the following: A:\>**TYPE MULTI.BAT** Enter



WHAT'S HAPPENING? You keyed in and saved a batch file called **MULTI.BAT** on the root of the DATA disk. You then displayed the contents of **MULTI.BAT** on the screen. To execute it you must not only key in the command name **MULTI** but must also provide the command with the positional parameters that are referred to in the file. First you will copy some files to the root of the DATA disk.

5 Key in the following: A:\>**COPY CLASS\VEN.*** Enter

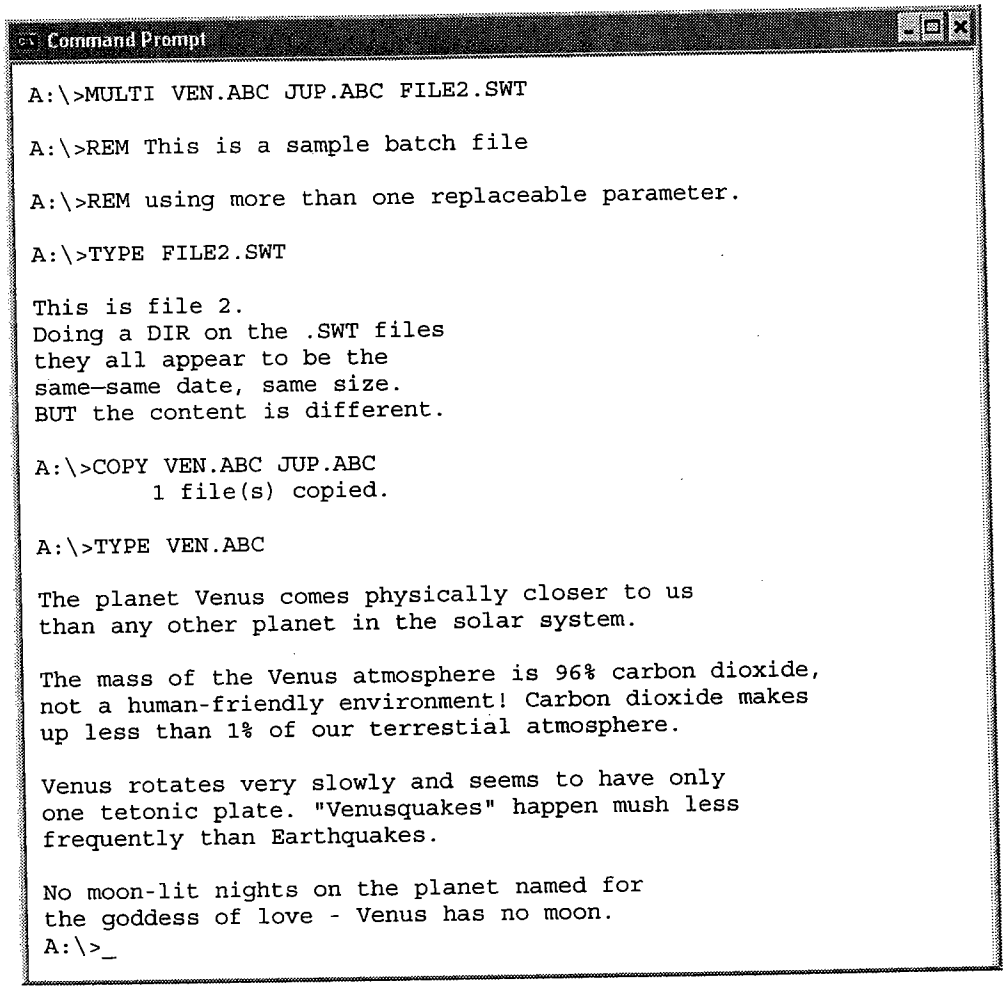


WHAT'S HAPPENING? You copied some files from the CLASS directory that had been moved from the root directory during disk organization (Chapter 8). In the next step, you will key in **MULTI VEN.ABC JUP.ABC FILE2.SWT**. The batch file knows what to put in each percent sign because it looks at the position on the command line. It does not matter which order you use the %1 or %2 or %3 in the batch file, only the order you use on the command line. See Table 10.5.

Position 0 on the Command Line	Position 1 on the Command Line	Position 2 on the Command Line	Position 3 on the Command Line
MULTI	VEN.ABC	JUP.ABC	FILE2.SWT
When the batch file needs a value for %0, it uses	When the batch file needs a value for %1, it uses	When the batch file needs a value for %2, it uses	When the batch file needs a value for %3, it uses
MULTI	VEN.ABC	JUP.ABC	FILE2.SWT
Command	Parameter	Parameter	Parameter

Table 10.5—Positional Parameters

6 Key in the following: A:\>**MULTI VEN.ABC JUP.ABC FILE2.SWT** Enter



WHAT'S HAPPENING? Each time the batch file came to a command line and needed a value for a replaceable parameter (**%1, %2,** or **%3**), it looked to the command line as it was keyed in by you, and it counted over (by position or location on the command line) until it found the value to replace for the percent sign. **MULTI.BAT** is actually in the first position, which is counted as %0. The command itself is always first, or %0. Thus, to indicate the position of the replaceable parameters, %1 refers to the first position after the command, not the first item on the command line. %2 refers to the second position after the command, not the second item on the command line,

and so on. Hence, when you refer to %1, you are referring to the first position after the command. When it needed a value for %1, it used **VEN.ABC** because that was in the first position on the command line. When it needed a value for %2, it used **JUP.ABC** because that was in the second position on the command line, and, when it needed a value for %3, it used **FILE2.SWT** because that was in the third position on the command line. Instead of calling them replaceable parameters, it is easier to remember them as positional parameters because it is the position on the command line that matters, not where it occurs in the batch file. Although this batch file may show you how the positional parameters work, it is not very useful. It does not accomplish any logical task. You are going to use the same principle to create a command that the operating system does not have.

7 Key in the following:

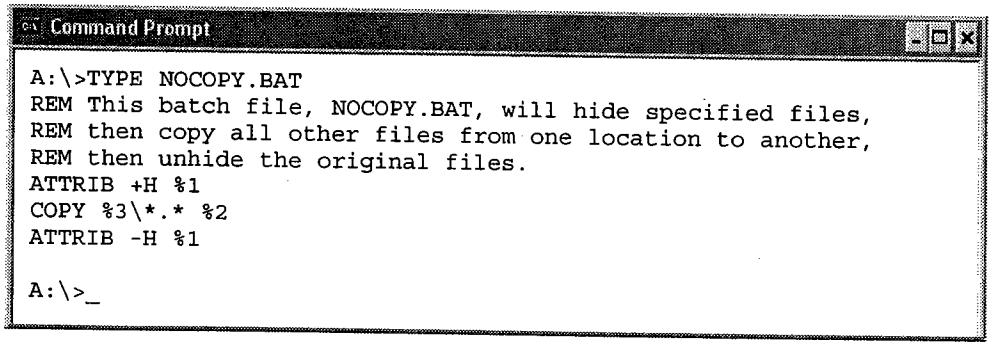
```
A:\>EDIT NOCOPY.BAT Enter
     REM This batch file, NOCOPY.BAT, will hide specified files, Enter
     REM then copy all other files from one location to another, Enter
     REM then unhide the original files. Enter
     ATTRIB +H %1 Enter
     COPY %3\*.* %2 Enter
     ATTRIB -H %1
```

WHAT'S HAPPENING? You created a batch file called **NOCOPY.BAT** using multiple positional parameters. You have created a command that the operating system does not have. It copies files selectively, allowing you to copy all files except those you hid. You must save the file to disk.

8 Press **Alt** + **F**. Press **X**.

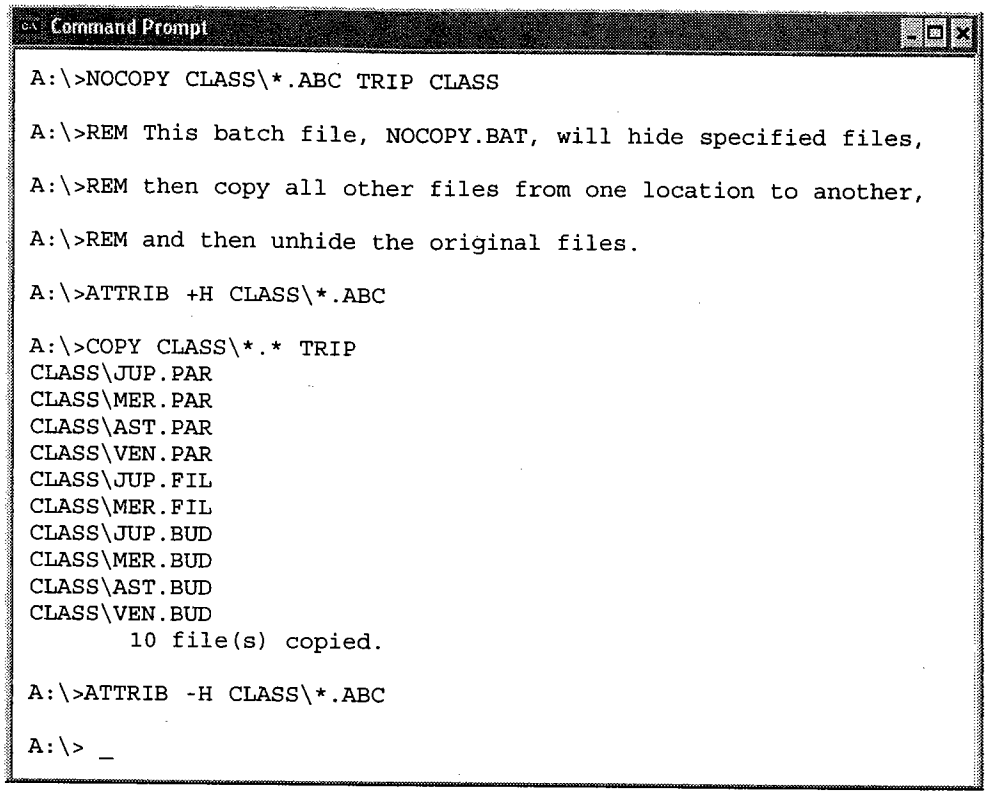
9 Press **Y**.

10 Key in the following: `A:\>TYPE NOCOPY.BAT` **Enter**



WHAT'S HAPPENING? You are displaying the contents of **NOCOPY.BAT**. To execute it, you must not only key in the command name—NOCOPY—but also provide the command with values for all the positional parameters. You want to copy all the files from the **CLASS** directory to the **TRIP** subdirectory except the files that have the **.ABC** file extension. Remember, parameters are separated by a space. On the command line you will key in **NOCOPY CLASS*.ABC TRIP CLASS**. The value *CLASS*.ABC* replaces parameter %1, *TRIP* replaces %2, and CLASS replaces %3. Notice that %1 will be used to represent a subdirectory and files ending with **.ABC**, while %2 and %3 will represent subdirectory names only.

11 Key in the following: `A:\>`**NOCOPY CLASS*.ABC TRIP CLASS** Enter

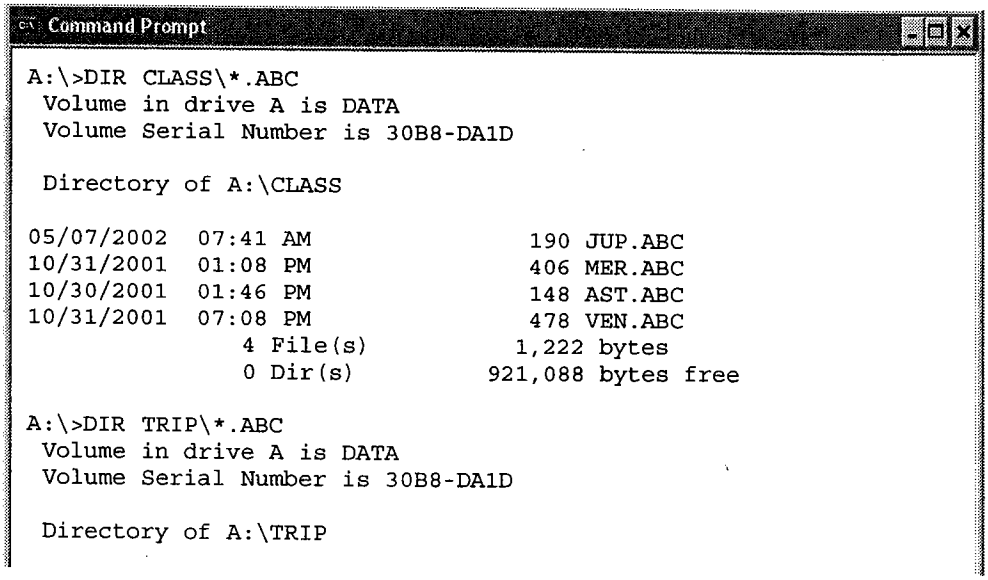


```
Command Prompt

A:\>NOCOPY CLASS\*.ABC TRIP CLASS

A:\>REM This batch file, NOCOPY.BAT, will hide specified files,

A:\>REM then copy all other files from one location to another,

A:\>REM and then unhide the original files.

A:\>ATTRIB +H CLASS\*.ABC

A:\>COPY CLASS\*.* TRIP
CLASS\JUP.PAR
CLASS\MER.PAR
CLASS\AST.PAR
CLASS\VEN.PAR
CLASS\JUP.FIL
CLASS\MER.FIL
CLASS\JUP.BUD
CLASS\MER.BUD
CLASS\AST.BUD
CLASS\VEN.BUD
       10 file(s) copied.

A:\>ATTRIB -H CLASS\*.ABC

A:\> _
```

WHAT'S HAPPENING? (*Note:* You may not see the command line you keyed in, as it may scroll off the screen.) You ran the batch file called **NOCOPY**. You substituted or provided the values: CLASS*.ABC (%1), TRIP (%2), and CLASS (%3). To check that the ***.ABC** files are not hidden in the **CLASS** directory and that they have not been copied to the **TRIP** directory, do the following:

12 Key in the following: `A:\>`**DIR CLASS*.ABC** Enter

13 Key in the following: `A:\>`**DIR TRIP*.ABC** Enter



```
Command Prompt

A:\>DIR CLASS\*.ABC
 Volume in drive A is DATA
 Volume Serial Number is 30B8-DA1D

 Directory of A:\CLASS

05/07/2002  07:41 AM               190 JUP.ABC
10/31/2001  01:08 PM               406 MER.ABC
10/30/2001  01:46 PM               148 AST.ABC
10/31/2001  07:08 PM               478 VEN.ABC
               4 File(s)          1,222 bytes
               0 Dir(s)         921,088 bytes free

A:\>DIR TRIP\*.ABC
 Volume in drive A is DATA
 Volume Serial Number is 30B8-DA1D

 Directory of A:\TRIP
```

```
File Not Found

A:\>_
```

WHAT'S HAPPENING? Your goal was achieved. To the operating system, the command sequence or string of commands looked like this:

ATTRIB +H CLASS*.ABC
COPY CLASS*.* TRIP
ATTRIB -H CLASS*.ABC

When you keyed in **NOCOPY CLASS*.ABC TRIP CLASS**, you asked the operating system to load the batch file called **NOCOPY.BAT**. The first position after **NOCOPY** has the value of **CLASS*.ABC**. The second position has the value of **TRIP**, and the third position has the value of **CLASS**. Then the lines were executed in order:

1. **REM This batch file, NOCOPY.BAT, will hide specified files,**
   This line is documentation for you to know why you wrote this batch file.
2. **REM then copy all other files from one location to another,**
   This line is a continuation of the documentation for you to know why you wrote this batch file.
3. **REM then unhide the original files.**
   This line is a continuation of the documentation for you to know why you wrote this batch file.
4. **ATTRIB +H CLASS*.ABC**
   This line tells ATTRIB to hide all the files in the **CLASS** directory with the file extension of **.ABC**. The operating system knew which file and which directory were %1 and could substitute CLASS*.ABC for %1 because CLASS*.ABC held the first position (%1) after the command NOCOPY.
5. **COPY CLASS*.* TRIP**
   This line tells the operating system to copy files in a directory. It knew in which directory to get the files because CLASS was %3, so it substituted CLASS for %3. The operating system knew it could substitute CLASS for %3 because CLASS was in the third position after NOCOPY. It knew to copy all the files because you included *.*. It knew which directory to copy the files to because it substituted TRIP for %2. It could substitute TRIP for %2 because TRIP was in the second position after NOCOPY.
6. **ATTRIB -H CLASS*.ABC**
   This line tells the operating system to unhide all the files in the **CLASS** directory with the file extension of **.ABC**. It knew which files and which directory were %1 and could substitute CLASS*.ABC for %1 because CLASS*.ABC held the first position after the command NOCOPY.

This command, which you just wrote as a batch file with replaceable parameters, can be very useful. You can use it to copy files selectively from one disk to another or from one subdirectory to another. You do not need to take separate steps because all the steps are included in the batch file.

10.30 Creating Useful Batch Files

Batch files are used to automate processes that otherwise take numerous commands in succession. With batch files you can, in essence, create new commands—commands that are not provided with the operating system. Or, as in the following example, re-create an old favorite command that isn't there anymore.

10.31 Activity: Writing Useful Batch Files

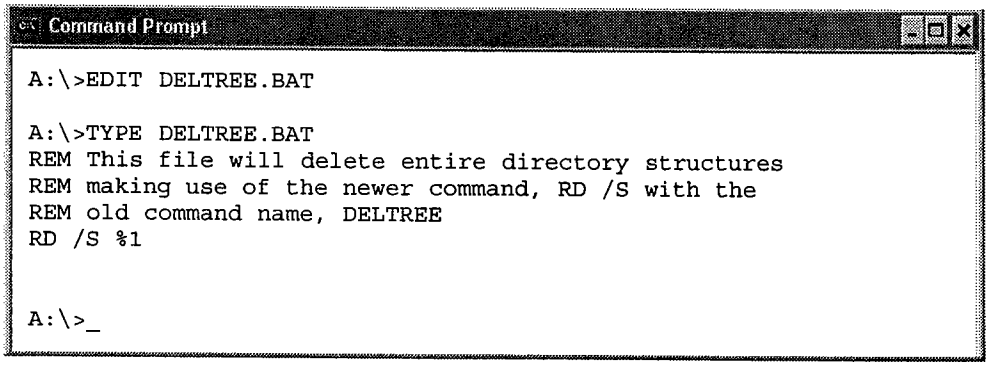
Note: The DATA disk is in Drive A with A:\> displayed.

1 Key in the following:
EDIT DELTREE.BAT Enter
REM This file will delete entire directory structures Enter
REM making use of the newer command, RD /S with the Enter
REM old command name, DELTREE Enter
RD /S %1

2 Press Alt + **F**. Press **X**.

3 Press **Y**.

4 Key in the following: A:\>**TYPE DELTREE.BAT** Enter

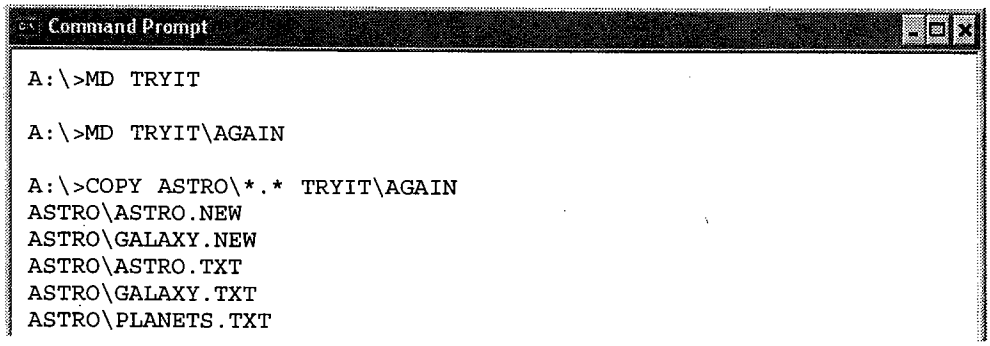


WHAT'S HAPPENING? You have written a file that re-creates the DELTREE command that "went away" with Windows 2000.

5 Key in the following: A:\>**MD TRYIT** Enter

6 Key in the following: A:\>**MD TRYIT\AGAIN** Enter

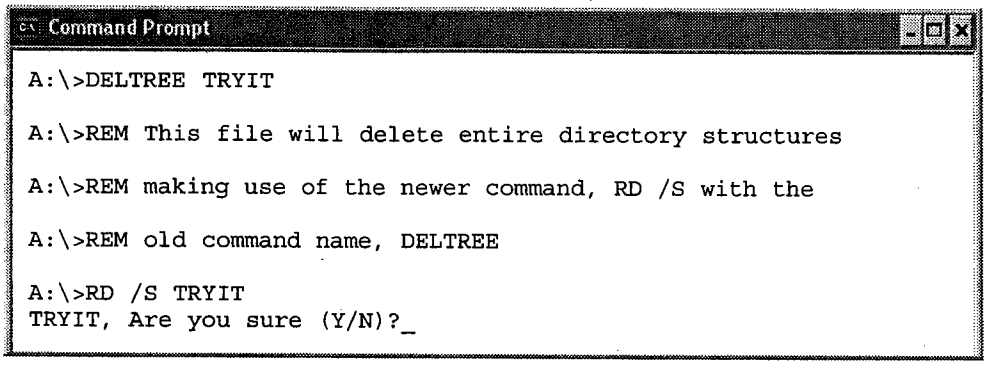
7 Key in the following: A:\>**COPY ASTRO*.* TRYIT\AGAIN** Enter



```
ASTRO\TITAN.TXT
ASTRO\AST.TST
ASTRO\AST.99
ASTRO\AST.BUD
ASTRO\ASTROLGY.FIL
ASTRO\ZODIAC.FIL
        11 file(s) copied.

A:\>_
```

WHAT'S HAPPENING? You have created a directory TRYIT that contains another directory AGAIN, and have copied some files into the AGAIN directory, in order to test your DELTREE.BAT file.

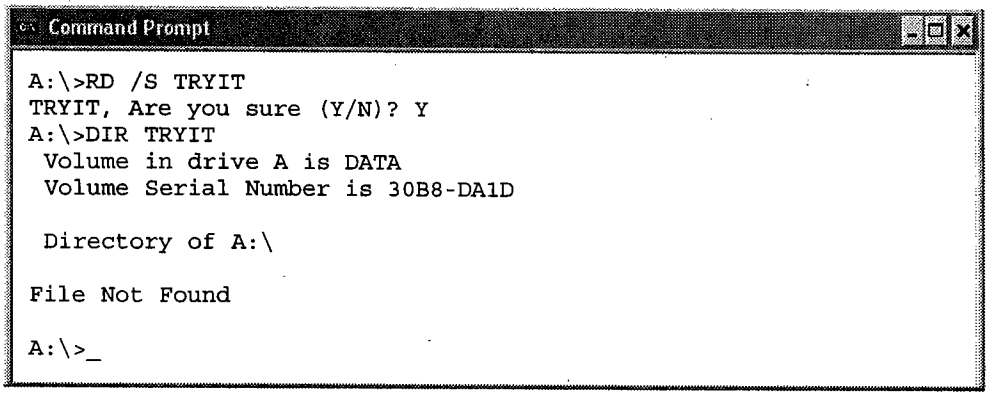
8 Key in the following: `A:\>`**DELTREE TRYIT** Enter



WHAT'S HAPPENING? The command on the fifth line, RD /S TRYIT, is being executed as if you had keyed it in at the command line. If on the last line you added redirection (RD /S %2 < Y.FIL), then you would not have to enter the Y. The batch file would not need user input. In this case, however, it does.

9 Key in the following: **Y** Enter

10 Key in the following: `A:\>`**DIR TRYIT** Enter



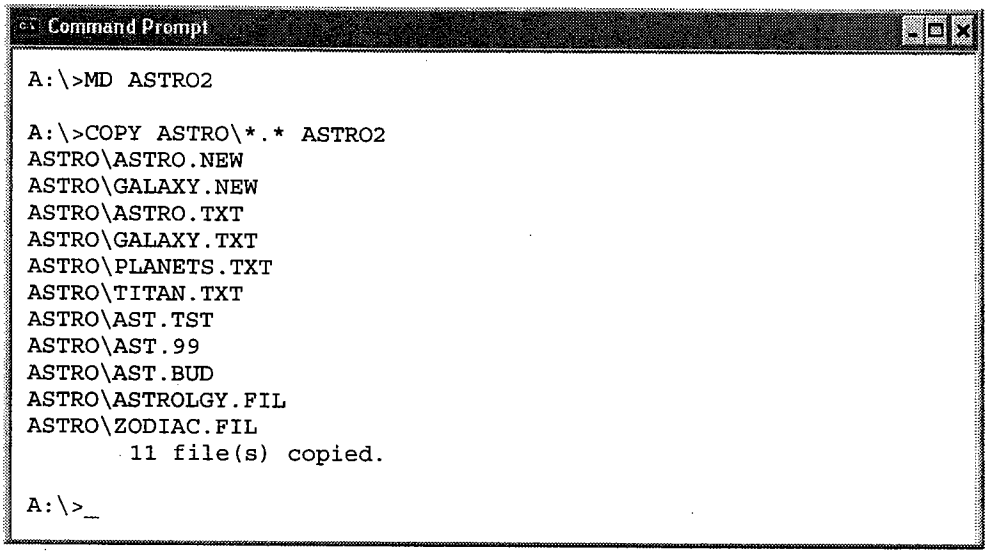
WHAT'S HAPPENING? The DELTREE batch file works just like the old DELTREE command used to work.

The next batch file you will write will solve a problem. For backup purposes, you often have the same files on more than one subdirectory or floppy disk. You may want to compare which files are in which directory or on which disk. This normally

would involve using the DIR command to view each directory or disk, or redirecting the output of the DIR command to the printer and comparing them. Why not let the computer do the work? You can do so with the FC command, which compares two files or sets of files and displays the differences between them. When creating a batch file, you should always test it on unimportant data. To be sure the batch file works correctly, we will set up a duplicate directory, compare it to the original, change the duplicate directories contents, and then compare it to the original again.

11 Key in the following: `A:\>`**MD ASTRO2** Enter

12 Key in the following: `A:\>`**COPY ASTRO*.* ASTRO2** Enter



WHAT'S HAPPENING? You have set up a scenario that enables you to check the viability of the batch file you are going to write.

13 Key in the following: `A:\>`
EDIT DCOMP.BAT Enter
REM This batch file will compare the file names Enter
REM in two directories. Enter
DIR /A-D /B /ON %1 > SOURCE.TMP Enter
DIR /A-D /B /ON %2 > OTHER.TMP Enter
FC SOURCE.TMP OTHER.TMP ¦ MORE Enter
PAUSE You are about to delete SOURCE.TMP and OTHER.TMP Enter
DEL SOURCE.TMP Enter
DEL OTHER.TMP

14 Press Alt + **F**. Press **X**.

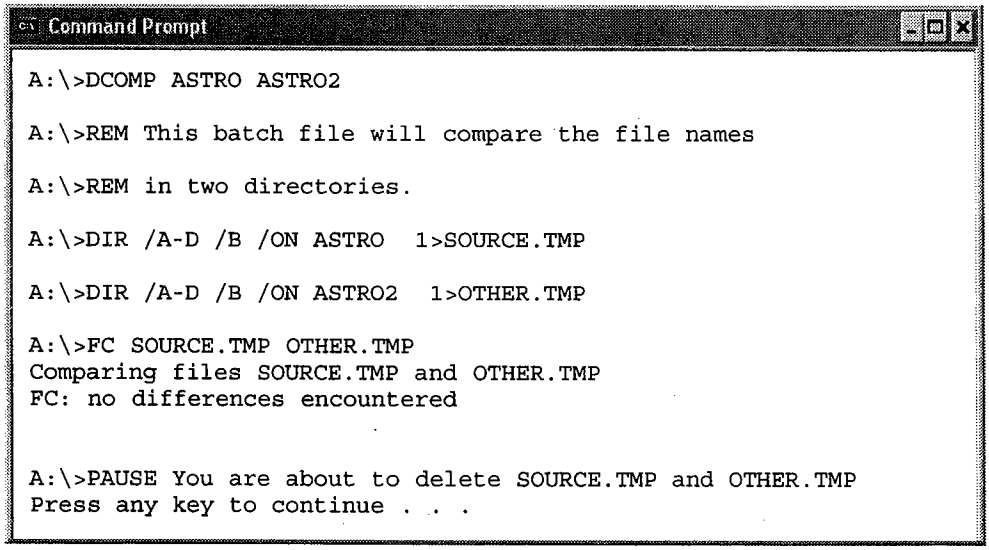
15 Press **Y**.

WHAT'S HAPPENING? This batch file uses the FC command to compare two files, each of which contains the file names in a different directory. You want to know if the same files are in each directory, and to do this, the files need to be displayed in the same order. Rather than writing the command line as **DIR ¦ SORT > SOURCE.TMP**, the command line was written as **DIR /A-D /B /ON %1 > SOURCE.TMP**. Why? You will examine each part of the line:

DIR /A-D	This is the command used to display files that do not have the directory attribute, so any subdirectory names will not be displayed.
/B	The /B parameter will eliminate all information in the directory display, such as **Volume in drive A is DATA** **Volume Serial Number is 3330-1807** **Directory of A:** as well as the file size, date, time, and long file name information.
/ON	This parameter will order the display by name, sorting the display alphabetically.
> SOURCE.TMP	This redirection character will send the output of the command line to the specified file, first **SOURCE.TMP**, and then **OTHER.TMP**. The files can now be compared to each other.

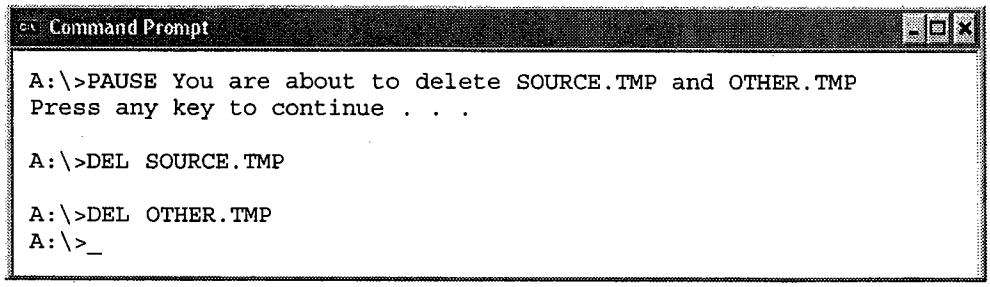
In addition, you took care to clean up by deleting both **SOURCE.TMP** and **OTHER.TMP** when the program finished executing. You are now going to test this batch file.

16 Key in the following: `A:\>`**DCOMP ASTRO ASTRO2** Enter



WHAT'S HAPPENING? The same files are in each subdirectory.

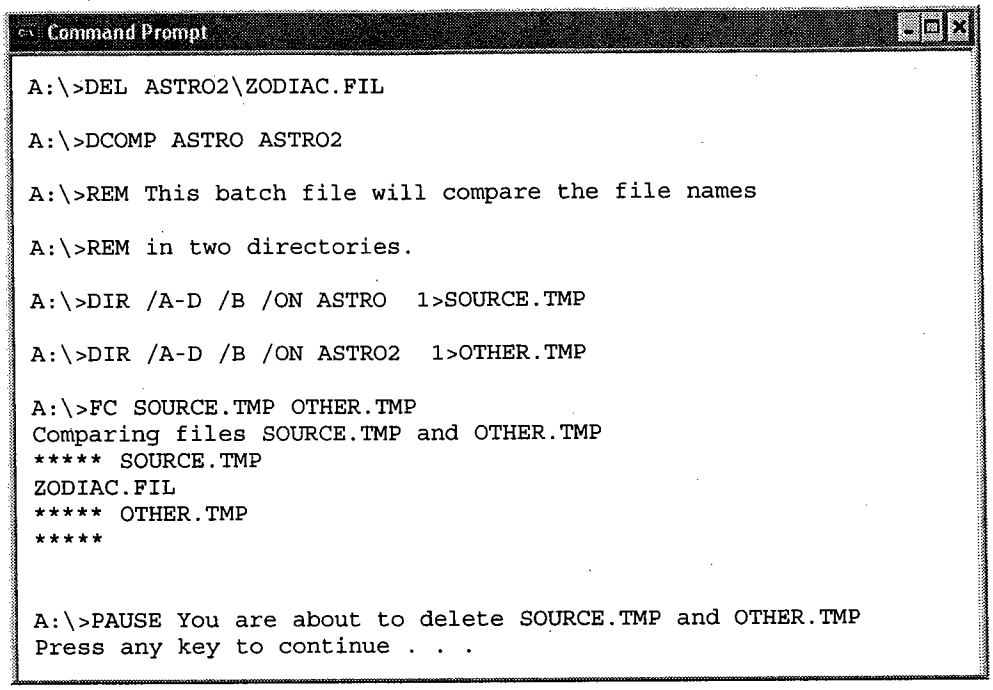
17 Press Enter



WHAT'S HAPPENING? You have deleted the **SOURCE.TMP** and **OTHER.TMP** files. What if there were differences?

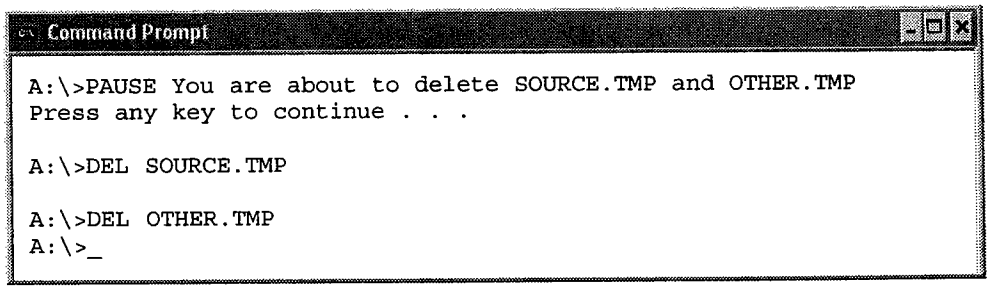
18 Key in the following: `A:\>`**DEL ASTRO2\ZODIAC.FIL** Enter

19 Key in the following: `A:\>`**DCOMP ASTRO ASTRO2** Enter



WHAT'S HAPPENING? There are differences. The **ASTRO** directory has one more file (**ZODIAC.FIL**) than the **ASTRO2** subdirectory.

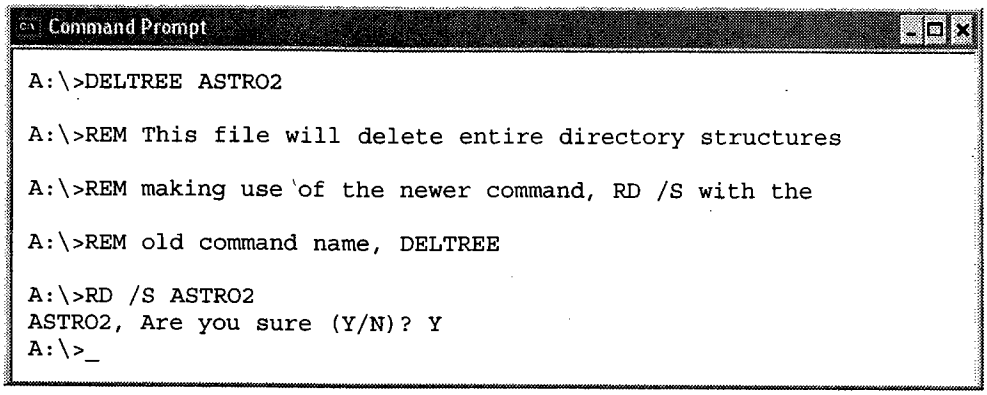
20 Press Enter



WHAT'S HAPPENING? You have deleted the temporary files. You now know you have a tested, working batch file that provides an easy way to compare the files in directories or on disks. You could add an @ECHO OFF statement at the beginning of the batch file so you would not see the REM statements and would see only the results of the command. You will use the DELTREE.BAT file to remove the ASTRO2 directory.

21 Key in the following: `A:\>`**DELTREE ASTRO2** Enter

22 Key in the following: **Y** Enter



WHAT'S HAPPENING? You have deleted the directory you created for testing purposes.

Chapter Summary

1. Batch processing means running a series of instructions without interruption.
2. Interactive processing allows the user to interface directly with the computer and update records immediately.
3. Batch files allow a user to put together a string of commands and execute them with one command.
4. Batch files have the .BAT or .CMD file extension.
5. Windows looks first internally for a command, then for a .COM file extension, then for a .EXE file extension, and finally for a .BAT or .CMD file extension.
6. Edit is a full-screen text editor used to write batch files.
7. A word processor, if it has a means to save files in ASCII, can be used to write batch files. ASCII files are also referred to as unformatted text files.
8. Batch files must be in ASCII.
9. A quick way to write an ASCII file is to use COPY CON. You copy from the console to a file.
10. Batch files are executed from the system prompt by keying in the batch file name.
11. Batch files are used for many purposes, such as to save keystrokes.
12. To "document" means to explain the purpose a file serves.
13. REM allows the user to document a batch file.
14. When the operating system sees REM, it displays on the screen whatever text follows REM. REM is not a command that executes.
15. ECHO OFF turns off the display of commands. Only the messages from the commands are displayed on the screen.
16. PAUSE allows the user to take some action before the batch file continues to execute.
17. PAUSE does not force the user to do anything. The batch file execution is suspended until the user presses a key.
18. To stop a batch file from executing, press the Ctrl key and the letter C (Ctrl + C).
19. Replaceable parameters allow the user to write batch files that can be used with many different parameters. The replaceable parameters act as place holders for values that the user will substitute when executing the batch file.

20. Replaceable parameters are sometimes called dummy, positional, or substitute parameters.
21. The percent sign (%) followed immediately by a numerical value, 0 to 9, indicates a replaceable parameter in a batch file.

Key Terms

batch file
batch processing
documented
dummy parameter
environmental variable
interactive processing
place holder
positional parameter
replaceable parameter
substitute parameter
variable

Discussion Questions

1. Explain the purpose and function of batch files.
2. Compare and contrast batch processing with interactive processing.
3. You have a batch file called CHECK.BAT. You key in CHECK at the prompt. Where does it look for the file? What does the operating system then do?
4. What is an ASCII file? Why is it important in batch processing?
5. Under what circumstances can a word processor be used to write batch files?
6. Compare and contrast using Edit and COPY CON to write batch files.
7. Explain the purpose and function of the /O and /A parameters when used with the DIR command.
8. Explain the purpose and function of the REM command. What happens when the operating system sees REM in a batch file?
9. In a data-processing environment, what does it mean to document a batch file? Why would it be important to document a batch file?
10. Explain the purpose and function of the ECHO command.
11. Explain the purpose and function of the PAUSE command.
12. Why does the PAUSE command require user intervention?
13. How can you stop a batch file from executing once it has begun?
14. What are parameters?
15. What is a replaceable parameter? Describe how it might be used.
16. What indicates to the operating system that there is a replaceable parameter in a file?
17. What advantages are there to using replaceable parameters in a batch file?
18. Replaceable parameters are sometimes called positional parameters. Explain.
19. There appear to be two prompts when you do not use the ECHO OFF. Explain.

True/False Questions

For each question, circle the letter T if the statement is true and the letter F if the statement is false.

T F 1. Any command that can be keyed in at the system prompt can be included in a batch file.
T F 2. Each command in a batch file is on a separate line.
T F 3. Batch files should be written to complete a process that will need to be done one time only.

T F 4. The PAUSE command, when used in a batch file, requires user intervention.

T F 5. In the batch file line COPY MYFILE YOURFILE, COPY would be %1.

Completion Questions

Write the correct answer in each blank space.

6. A word-processing program can be used to write batch files if it has a(n) _____ output mode.
7. The maximum number of replaceable parameters in a batch file is _____.
8. In a data-processing environment, to explain the purpose of a batch file is to _____ it.
9. When you press the Ctrl and C keys simultaneously during the execution of a batch file, you will see _____ on the screen.
10. If you wanted only the output of the command displayed on the screen and not the command itself, you would begin the batch file with the line _____.

Multiple Choice Questions

For each question, write the letter for the correct answer in the blank space.

____ 11. When searching a disk for an external command, the operating system will
   a. choose the command with the extension .COM before one with the extension .BAT.
   b. choose the command with the extension .BAT before one with the extension .COM.
   c. know automatically which file extension you are seeking.
   d. choose the extension .BAT before .EXE.

____ 12. To execute a batch file called **THIS.BAT** at the prompt, key in:
   a. THIS %1
   b. %1
   c. THIS
   d. RUN BATCH THIS

____ 13. In a batch file, to display the output of the command but not the command itself, use:
   a. ECHO ON
   b. ECHO OFF
   c. DISPLAY ON
   d. none of the above

____ 14. Which of the following statements is true?
   a. Batch files cannot use variable parameters.
   b. Ctrl + Z interrupts the execution of a batch file.
   c. The PAUSE command will wait until the user takes some action.
   d. either a or b

15. Quitting a batch file by using Ctrl + C
    a. erases all the lines of the batch file that come after you quit.
    b. can be done only at a PAUSE during the batch file run.
    c. erases the batch files from the disk.
    d. can be done at any time while the batch file is running.

Writing Commands

Write five lines that would perform the following items in a batch file. Each question represents one line in the file.

16. Document the batch file, explaining that it is a demonstration file.

17. List the files in the **A:\TEMP** directory.

18. Display the contents of a file in the **A:\TEMP** directory that is specified when the batch file is called.

19. Rename the above file to **NAME.NEW**.

20. Display the contents of the **NAME.NEW** file.

Homework Assignments

Problem Set I

Note 1: Place the HOMEWORK disk in Drive A. Be sure to work on the HOMEWORK disk, not the DATA disk. On each batch file you write, be sure and include *your name*, the *name of the batch file*, and the *date* as part of the documentation.

Note 2: The homework problems will assume Drive C is the hard disk and the HOMEWORK disk is in Drive A. If you are using another drive, such as floppy drive B or hard drive D, be sure and substitute that drive letter when reading the questions and answers.

Note 3: Test all of your batch files before submitting them to be sure they work correctly.

Note 4: To save a file with Edit under a new name, press Alt + F and choose **Save As**.

Note 5: It will be assumed that the root of the HOMEWORK disk is the default drive and directory, unless otherwise specified.

Note 6: There can be more than one way to write a batch file. If your batch file works correctly, it is most likely written correctly.

Problem A

To Create DD.BAT

A-a Create a batch file called **DD.BAT**. This batch file should display the files on the root of the HOMEWORK disk in date order with the most current date displayed first, and should pause so that the user may view the display one screenful at a time.

A-b Document **DD.BAT**. Remember to include *your name*, the *name of the batch file*, and the *date* as part of the documentation.

To Create DDA.BAT

A-c Edit the batch file called **DD.BAT** and save it under the new name of **DDA.BAT**.

A-d Edit **DDA.BAT** so that it will look on any drive for files, using a replaceable parameter. It will still be displaying files in date order, with the most current date displayed first.

A-e Update the documentation.

To Create SD.BAT

A-f Create a batch file called **SD.BAT** that will display subdirectories only, in order by name, on the root of the A drive.

A-g Document the file. Remember to include your name, the name of the batch file, and the date as part of the documentation.

To Create SDD.BAT

A-h Edit **SD.BAT** created above and save the edited file as **SDD.BAT**. The **SDD.BAT** file will display subdirectories on a specified drive\directory in order by name.

A-i Use replaceable parameters.

A-j Document the file.

To Print Your Homework

1 Be sure the printer is on and ready to accept print jobs from your computer.

2 Key in the following: `A:\>`**NAME** Enter

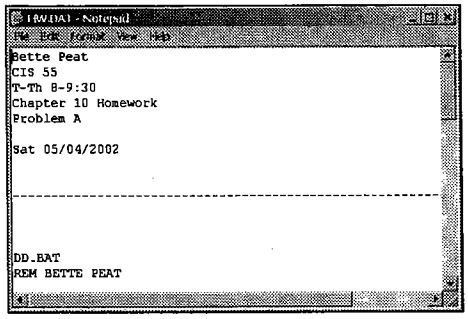
3 Here is an example to key in, but your instructor will have other information that applies to your class. Key in the following:
Bette A. Peat Enter (*Your* name goes here.)
CIS 55 Enter (*Your* class goes here.)
T-Th 8-9:30 Enter (*Your* day and time go here.)
Chapter 10 Homework Enter
Problem A Enter

4 Press F6 Enter

5 If the information is correct, press **Y** and you are back to A:\>.

6 Key in the following:
GO NAME.FIL DD.BAT DDA.BAT SD.BAT SDD.BAT Enter
(These files will not be deleted from your disk after printing.)

7 Press **N** until you see the Notepad window.



8 In Notepad, click **File**. Click **Print**. Click the **Print** button. Close Notepad.

Problem B

To Create EXTRA.BAT

B-a Copy all the files with the **.TXT** extension from the **WUGXP** subdirectory to the root directory of the HOMEWORK disk. (Overwrite existing files.)

B-b Create a batch file named **EXTRA.BAT** that will do the following:

- Clear the screen.
- Display the root directory of the HOMEWORK disk for any file that has **.TXT** as a file extension.
- Make a copy of the file called **BORN.TXT** and call the copy **BIRTH.STR**.
- Display the contents of the file called **BIRTH.STR**.
- Give the user time to read the file.
- Erase the file called **BIRTH.STR**.

B-c Document **EXTRA.BAT**, including your name, the name of the file, and the date.

To Create EXTRA2.BAT

B-d Edit **EXTRA.BAT**. **EXTRA2.BAT** will do all that **EXTRA.BAT** does but will

- use replaceable parameters so that the user may choose what files to display. (Use only one parameter to represent the entire file specification, such as *.fil or my.fil.)
- allow any existing file to be copied to a new file that the user names.
- display the contents of the newly created file.
- provide a way for the user to change her or his mind and not delete any files.

B-e Save the edited file as **EXTRA2.BAT**.

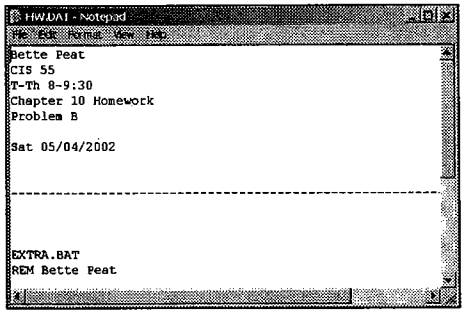
To Print Your Homework

1 Be sure the printer is on and ready to accept print jobs from your computer.

2 Use Edit to edit **NAME.FIL**. Change the last line from **Problem A** to **Problem B**.

3 Key in the following: **GO NAME.FIL EXTRA.BAT EXTRA2.BAT** Enter

4 Press **N** until you see the Notepad screen.



5 In Notepad, click **File**. Click **Print**. Click the **Print** button. Close Notepad.

Problem C

To Create CHECKIT.BAT

C-a Write and document a batch file using replaceable parameters called **CHECKIT.BAT** that will:

- check the status of a specified disk.
- see if any files are noncontiguous on that disk.
- do a wide display of the root directory of that disk.
- pause as necessary so users can view all of each display.

To Create LIST.BAT

C-b Write and document a batch file called **LIST.BAT** that will use replaceable parameters to display the contents of three files. (*Hint*: Think about how many replaceable parameters you will want to use.) Use **JUP.99**, **BORN.TXT**, and **COMEDY.TV** to test your file.

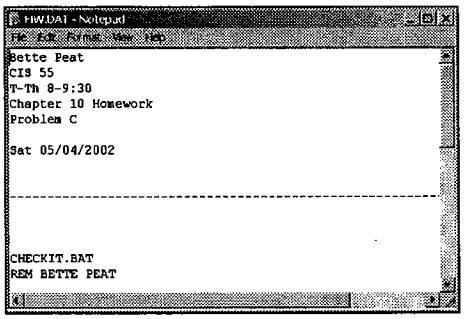
To Print Your Homework

1 Be sure the printer is on and ready to accept print jobs from your computer.

2 Use Edit to edit **NAME.FIL**. Change the last line from **Problem B** to **Problem C**.

3 Key in the following: **GO NAME.FIL CHECKIT.BAT LIST.BAT** Enter

4 Press **N** until the Notepad screen appears.



5 In Notepad, click **File**. Click **Print**. Click the **Print** button. Close Notepad.

Problem D: Challenge Assignments

The following three batch file assignments use commands in combination that may not have been specifically discussed or shown by example in the chapter. Research and experimentation may be required in order to write these batch files.

To Create NODEL.BAT

D-a Using **NOCOPY.BAT**, created in this chapter, as a model, write and document a batch file called **NODEL.BAT** that will

- allow you to delete all the files *except* the file(s) you hide in a specified subdirectory.
- view the files you are about to delete.
- clear the screen when necessary to make it easy for the user to read what is happening in the file.
- give the user an opportunity to cancel prior to deleting the files.
- delete the files.
- unhide the files.
- view which files remain.

D-b Document the batch file completely, explaining precisely what the file will do, and giving an example of how to run the file.

Hint 1: Create a practice subdirectory, such as **\TEMP**, on the HOMEWORK disk. Use the **NODEL** batch file with the files that you copy into this temporary subdirectory, such as the ***.RED** files and the ***.FIL** files. If you make a mistake, you will not delete all the files on your HOMEWORK disk.

Hint 2: If you do not want to be asked for confirmation from the DEL command, you can use the command DEL *.* < Y.FIL. You will need to copy **Y.FIL** from **\WUGXP** to the root directory of the HOMEWORK disk.

To Create NEWSUB.BAT

D-c Write and document a batch file called **NEWSUB.BAT**, using replaceable parameters, that will accomplish what is specified in the following documentation:

```
REM This file will allow the user to create a subdirectory
REM on a drive and with a name specified by the user
REM and then copy files specified by the user from a location
REM specified by the user to the new subdirectory.
REM This batch file is flexible, so any of the elements can be
REM changed by the user when they execute this file.(The 4 elements
REM are entered separately on the command line by the user, and are
REM DRIVE NAME (such as A:), the NEW SUBDIRECTORY NAME,
REM the LOCATION OF FILES TO BE COPIED (such as C:\WUGXP), and
REM the names of the FILES TO BE COPIED.)
REM The newly created subdirectory with its files will then be
REM displayed on the screen. Be sure the user has the opportunity to
REM view all of the display. The command line would read something
REM like    NEWSUB  A:  TEMP  C:\WUGXP  *.99
```

D-d Include the above lines in the documentation of the file.

D-e You may want to delete the **TEMP** directory created with the **NODEL.BAT** assignment, and use it again to write and debug **NEWSUB.BAT**.

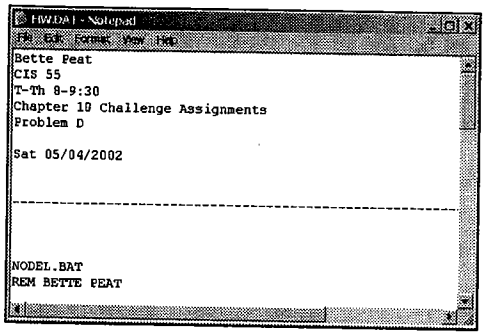
To Print Your Homework

1 Be sure the printer is on and ready to accept print jobs from your computer.

2 Use Edit to change the last two lines in **NAME.FIL** to read:
Chapter 10 Challenge Homework
Problem D

3 Key in the following: **GO NAME.FIL NODEL.BAT NEWSUB.BAT** Enter

4 Press **N** until you see the Notepad screen.



5 In Notepad, click **File**. Click **Print**. Click the **Print** button. Close Notepad.

CAUTION: If you created a TEMP directory on the HOMEWORK disk, delete that directory before going on to Chapter 11.

Problem Set II—Brief Essay

1. Assume the following scenario:
   On your computer, you have many text files in the C:\MYSTUFF\DOSTEXT subdirectory. You refer to these files frequently at the command line, and you do not want to have to key in the path name each time you want to access one of the files. Instead, you would like to use the unused drive letter F to access this drive. Explain, in detail, how you would go about making the F drive specification available to you each and every time you opened a command line session.
2. Discuss replaceable parameters, including how they work and why they are useful. Include the number of parameters available and the numbering sequence used to represent the command in your discussion.

CUSTOMER'S POSTING RECEIPT

Post Type:	Stored Value
Module:	Dining Services
Receipt Number:	49881
Customer Name:	GUEST
Card Number:	00000000000001000007813
Workstation POS:	JARMSTRONG
Profit Center:	STUDENT ACCOUNTS
Fund Name:	PRINT $
Trans. Type:	Payment
Tender Type:	Cash 1
Amount:	$1.00
Trans. Entered:	01/21/2009 02:33 PM
Trans. Accepted:	01/21/2009 02:33 PM