New Version 3.3

DOS

the *EASY* Way

A Complete Guide to Microsoft's MS-DOS

Everett Errol Murdock, Ph.D.

SOE8D-INTERESC-CSUF
05119

DOS the *EASY* Way

A Complete Guide to Microsoft's MS-DOS

Everett Errol Murdock, Ph.D.

- Operating System Basics
- Tips for Beginners
- Tips for Advanced Users
- How to Use Fixed Disks
- How to Create Batch Files
- DOS Command Reference
- Version 3.3 Commands

H.O.T. Press
362 Miraleste Dr. #356
San Pedro, CA 90732.

Cover design by EZ Technical Systems

Editorial and production supervision: Susan Sudbury

IBM and IBM-PC are trademarks of International Business Machines Corporation

Microsoft, MS, and MS-DOS are trademarks of Microsoft Corporation

This document contains the latest information available at the time of publication. Everett Murdock and H.O.T. Press reserve the right to modify and revise the contents of this document at any time. No contractual obligation is expressed or implied.

This document is copyrighted material and may not be copied or reproduced in any way or stored or transmitted in an electronic medium without the express written consent of Everett Murdock and H.O.T. Press.

Printed in the United States of America

Copyright © 1988 by Everett Errol Murdock

EasyDOS Companion Disk

A companion DOS training and HELP program is also available. The *EasyDOS* Companion Disk includes on-line help for each of the DOS commands and configuration options. In addition, all of the programs and batch files described in this book are included on disk.

The *EasyDOS* program and the batch files are available from H.O.T. Press at a cost of $19.95 plus $3.00 for mailing and handling (California residents should also include $1.30 sales tax).

This book is available for distribution with hardware systems or other volume uses. It can be modified to suit users that order volume quantities. For more information, contact H.O.T. Press at (213) 831-1473.

H.O.T. Press
362 Miraleste #356
San Pedro, CA 90732.

Preface

This book is about a set of computer operating programs collectively known as a disk operating system (often abbreviated as DOS). The particular disk operating system described in this book is the MS-DOS system developed and sold by the Microsoft Corporation; it is used on dozens of different brands of personal computers. Millions of these computers have been sold and, because they are similar to the personal computers manufactured and sold by the IBM corporation, they are often referred to as "IBM-compatible." IBM-compatible computers vary widely in price, appearance, and function; however, they all have one thing in common: they all use the same disk operating system, the MS-DOS set of operating programs.

This book provides operating system concepts for both beginners and advanced users, as well as a guide to the complete set of commands provided on the first three versions of DOS. All DOS programs and capabilities implemented through version 3.3 are explained in detail.

About Ease-of-Use

This book is designed to be as easy to use as its title implies. It makes no assumtions about the expertise of the reader, yet it progresses step-by-step from the basic DOS concepts to some very advanced ones.

In general, the first three chapters are introductory: little experience with computers or operating systems is required to master the concepts in these chapters.

With the background information provided in the first three chapters in hand, inexperienced readers may venture into the more complex realms of batch files and advanced commands presented in chapters four and five. Even here, the reader is not left to struggle through these concepts; each new topic is explained in detail and a number of carefully-described tryout examples are presented along with each new concept.

Chapter 6, the DOS command reference, is designed to provide both a quick reference and a detailed description of every DOS command. It provides not only complete descriptions of each command, but explanations of their function and purpose. It tells you why you might want to use the command. Most importantly, it provides useful examples of each of the DOS commands.

The method of presentation used in this book relies heavily on tryout examples. Every chapter provides real-life, useful examples: you can prop up this book next to your computer and actually try out the commands. There is no better way to learn how to use DOS.

Chapter Summary

Chapter 1. Introduction

Chapter 1 provides an introduction to operating system concepts in general, and to Microsoft's disk operating system in particular. It describes startup procedures and introduces both the internal and external DOS programs and command names. Many examples are provided – useful procedures that you can try out using your computer.

This chapter also presents a complete description of a DOS format notation system that will make using the DOS commands easier. Additional sections describe the basic command components, using DOS filenames, using wildcards, and disk types and disk usage.

Chapter 2. Using Common DOS Commands

Chapter 2 focuses on common DOS problems and describes the most frequently used DOS commands. A "basic set" of useful DOS commands is described in detail. They are:

CHKDSK	DISKCOPY	RENAME (REN)
COPY	ERASE (DEL)	TIME
DATE	FORMAT	TYPE
DIR	PRINT	

Chapter 3. Using Fixed Disks

Chapter 3 provides an introduction to the use of computer systems that include one or more fixed disks (sometimes referred to as "hard" disks).

The DOS commands most commonly used with fixed-disk systems are discussed in detail. They are:

MKDIR (MD)	PATH
RMDIR (RD)	APPEND
CHDIR (CD)	TREE

Knowledge of these commands are a necessity for anyone planning to use DOS directories and subdirectories on a fixed disk system. Special commands that are commonly provided with these types of systems to prepare and format the fixed disk are also discussed.

Chapter 4. Using Batch Files

Chapter 4 introduces the many functions and purposes of batch files, with complete instructions on how to create and use them to carry out many of your computer tasks. This chapter also describes a very special batch file, the AUTOEXEC batch file.

Advanced topics include passing parameters to batch files and creating user-choice menus with batch "programs." Tips are provided on creating interactive batch files and taking input from the batch file user.

The complete set of DOS batch commands is described in detail. They are:

CALL	GOTO	REM
ECHO	IF	SHIFT
FOR	PAUSE	

Chapter 5. Tips for Advanced Users

Chapter 5 describes a variety of advanced techniques useful to any user ready to learn more about DOS. Some of the topics described are:

- Using ASCII characters
- Using the DOS editing keys
- DOS redirection and piping
- Laser printers
- The GRAPHICS command
- DOS device names
- DOS filter commands
- DOS printing methods

A special section describes system configuration issues, including using the CONFIG.SYS file. The complete set of DOS system configuration commands are described in detail. They are:

BREAK	DEVICE	LASTDRIVE
BUFFERS	FCBS	SHELL
COUNTRY	FILES	STACKS

A large number of examples are provided. These are working examples that can be typed in and tried out on your computer. Also included are a number of useful batch programs that can be typed in and used as they are, or they can be modified for your own needs.

A special set of advanced DOS commands are discussed in detail. They are:

MODE
PROMPT
SET
XCOPY

Chapter 6. DOS Command Reference

Chapter 6 is a complete reference to all of the DOS commands. The chapter begins with a quick reference summary showing the syntax and purpose of each of the commands. Then the commands are presented alphabetically: each is described in detail with tryout examples.

Appendixes

The appendixes provide information on DOS error messages, on country-specific information, and on the ASCII character set.

Contents

Chapter 1. Introduction

Operating Systems 1-1
Microsoft's Disk Operating System 1-2
Startup 1-3
The DOS Prompt 1-4
DOS Program Types 1-5
Internal Programs 1-5
External Programs 1-8
DOS Format Notation 1-10
Notation Summary 1-11
DOS Notation Rules 1-13
Notation Example: The DIR Command 1-14
DOS Filenames 1-16
Designing Useful Filenames 1-19
Using Wildcards 1-19
Using the Question Mark in Filenames 1-19
Using the Asterisk in Filenames 1-20
Disks and File Storage 1-21
Diskette Cautions 1-21
Disk Types 1-22
Disk Storage Capacity 1-24
DOS Error Messages 1-25

Chapter 2. Using Common DOS Commands

Frequently Used DOS Commands 2-2
CHKDSK 2-2
COPY 2-5
DATE 2-8
DIR 2-9
DISKCOPY 2-9
DEL (ERASE) 2-11

FORMAT 2-12
PRINT 2-16
RENAME 2-20
TIME 2-21
TYPE 2-22
Other Useful Commands 2-23

Chapter 3. Using Fixed Disks

Using DOS Directories 3-1
Special Fixed Disk Usage Commands 3-4
MKDIR (MD) 3-4
RMDIR (RD) 3-6
CHDIR (CD) 3-7
PATH 3-8
APPEND 3-10
TREE 3-11
Setting Up a Fixed Disk 3-12

Chapter 4. Using Batch Files

What are Batch Files? 4-1
Creating a Batch File 4-2
The AUTOEXEC.Batch File 4-3
Passing Parameters to Batch Files 4-5
Batch Command Summary 4-9
CALL 4-9
ECHO 4-10
FOR 4-11
GOTO 4-13
IF 4-15
SHIFT 4-19
PAUSE 4-20
REM 4-21
Advanced Batch Techniques 4-22
Useful Batch Files 4-22
Advanced Batch Files 4-24
Interactive Batch Files 4-27
Creating GET.COM 4-27

Creating Menus 4-29
Getting Yes/No Input 4-32
A Powerful AUTOEXEC.BAT File 4-33
Creating Additional Batch Files 4-37

Chapter 5. Tips for Advanced Users

The Advanced DOS User 5-2
Using ASCII Characters 5-2
Using the DOS Editing Keys 5-4
DOS Device Names 5-7
DOS Filter Commands 5-8
DOS Redirection and Piping 5-8
Redirecting DOS Input and Output 5-9
Redirection Examples 5-9
DOS Command Piping 5-15
Other Redirection Techniques 5-18
Configuring Your System 5-19
CONFIG.SYS 5-19
BREAK 5-20
BUFFERS 5-20
COUNTRY 5-21
DEVICE 5-22
FCBS 5-22
FILES 5-23
LASTDRIVE 5-24
SHELL 5-24
STACKS 5-25
DOS Printing Methods 5-25
Print Program Options 5-26
Other Printing Methods 5-27
Special Printer Capabilities 5-27
A Sample Printer Program in BASIC 5-28
A Batch File Printer Program 5-31
Simple Printer Batch Files 5-34
Printer Control Using PROMPT 5-34
Using Laser Printers 5-36
Using the GRAPHICS Command 5-37
Advanced DOS Commands 5-37
MODE 5-38
PROMPT 5-46
SET 5-56
XCOPY 5-58

Chapter 6. DOS Command Reference

DOS Command Syntax Notation 6-1
Command Syntax Elements 6-2
Command Usage 6-3
DOS Command Summary 6-5
DOS Command Reference 6-12
APPEND 6-12
ASSIGN 6-15
ATTRIB 6-17
BACKUP 6-20
BREAK 6-23
CALL 6-24
CHCP 6-25
CHDIR (CD) 6-27
CHKDSK 6-29
CLS (Clear Screen) 6-33
COMMAND 6-34
COMP 6-36
COPY 6-38
CTTY 6-41
DATE 6-42
DEL 6-43
DIR 6-44
DISKCOMP 6-46
DISKCOPY 6-47
ECHO 6-48
EXE2BIN 6-49
EXIT 6-51
FASTOPEN 6-52
FC 6-53
FDISK 6-55
FIND 6-56
FOR 6-58
FORMAT 6-59
GOTO 6-61
GRAFTABL 6-62
GRAPHICS 6-64
IF 6-66
JOIN 6-70
KEYB 6-72
LABEL 6-74
MKDIR (MD) 6-75

MODE 6-76
MORE 6-84
NLSFUNC 6-86
PATH 6-87
PAUSE 6-88
PRINT 6-89
PROMPT 6-92
RECOVER 6-94
REM 6-95
RENAME 6-96
REPLACE 6-97
RESTORE 6-99
RMDIR 6-101
SELECT 6-102
SET 6-104
SHARE 6-106
SHIFT 6-107
SORT 6-109
SUBST 6-111
SYS 6-113
TIME 6-115
TREE 6-117
TYPE 6-118
VER 6-119
VERIFY 6-120
VOL 6-121
XCOPY 6-122

Appendix A. DOS Error Messages
Appendix B. Country-Specific Information
Appendix C. ASCII Character Table

Chapter 1
Introduction

Many people have problems using computers. Some have a real fear of them. But they soon learn that it's not the computer itself that is hard to deal with, it's the "language" that they have to learn in order to "talk" to the computer.

A computer is a made up of wires and electronic components; that's why it is known as "hardware." But, the computer cannot act without the other half of its operational tools: the programs that are loaded into the computer. These programs are known as the computer's "software." When you use a personal computer, you are actually interacting with the functions provided by the software, not the hardware. And the first set of software programs you must learn to use are the computer's operating programs; it can safely be said that if you know how to use a computer's operating programs, you know how to use the computer.

This chapter is an introduction to the disk operating system software used in today's personal computers: it describes what an operating system is and summarizes what you can do using operating system commands.

Operating Systems

There are many types of software programs that can be used on personal computers, but before any of them can function, a set of software control programs must first be called into action. These programs are collectively known as the operating system.

The most important software for any computer system is its operating system. Without an operating system, a computer is an inert set of wires and circuits just waiting to go into action. It is the operating system that starts the action.

The operating system (often referred to as DOS, or Disk Operating System) provides a way to interact with your computer: you give your computer operating commands by using the programs that are part of the operating system.

Microsoft's Disk Operating System

The operating system used with a great many personal computers is the set of operating programs developed by the Microsoft Corporation. Referred to as the MS-DOS operating system, these programs are usually provided on two diskettes. They are collectively known as MS-DOS, PC-DOS (the IBM PC version), or just DOS. DOS, in each case, stands for Disk Operating System.

DOS provides a way for you to interact with the computer by giving you a large set of operating programs, stored on disk. Each one of these small operating programs is used to carry out a computer task. You activate a program by entering its name: the name you use to activate a DOS program is known as a DOS command.

Some of the things you can do using the DOS commands are:

- Tell the computer how to handle incoming information.
- Tell the computer how and where to store information.
- Tell the computer how to display information on the screen.
- Tell the computer how to interact with peripheral devices such as printers and disk drives.

In general, you use DOS programs to give your computer instructions. The trouble is, in order to give the computer accurate instructions, you have to know exactly how to use each of the DOS commands. Many of the DOS commands are very simple to use and do obvious things. But some of them have complicated options that are not so simple. This book will show you how to use the DOS commands, including all of the command options. We will try to clear up some of the DOS mysteries by describing, in detail, the nature and purpose of each of the DOS command programs.

Startup

When you first start up your personal computer – either by using the DOS startup disk or by using DOS command programs that have been transferred to a fixed disk – the computer displays the startup screen. It should look similar to this:

```
MS-DOS Version 3.3
Copyright 1981-1988 Microsoft Corp.

Current date is Fri 6-17-1988
Enter new date (mm-dd-yy):
```

As indicated by this example of an opening screen, on most computers DOS first asks you to enter the date and time. Some computers skip this step and automatically start up other types of programs.

After you enter the correct date (use exactly the same format as indicated in the current date), the operating program will next ask you to enter the time.

Next, the following prompt will be displayed:

```
A>
```

The letter A followed by the right angle bracket (>) shown above is known as the "DOS prompt." It is usually displayed at the left side of the computer's screen at the position where you can start entering data.

The DOS Prompt

The DOS prompt is an indicator that the system is ready to go to work. If, as in the example above, the computer was started up by turning it on with the DOS program disk in drive A, DOS first loads the necessary operating programs into the computer's memory and then displays the letter A to indicate that it is ready to go to work. The letter A indicates that the active disk drive is drive A.

Everyone who has used a computer with the MS-DOS operating system, at first wondered, "What is that A doing on the screen? What am I supposed to do now?"

The answer, you soon learn, is that it is waiting for you to enter the name of a program so it can load that program and begin to operate under the guidance of the program's instructions. The program name you enter can be the name of a DOS operating program or any other program that has been developed expressly for your computer. Either way, you call the computer into action by typing in the program name right next to the DOS prompt and then pressing the **Enter** key (sometimes referred to as the **Return** key).

If you want to use a program on a diskette in another drive, it's easy to change the active drive. Just enter the letter designating the drive, followed by a colon. For example, if you entered "B:" right next to the A prompt and then pressed the Enter key, the DOS prompt would immediately change to

```
B>
```

The operating system would now look on the diskette in drive B for any program name you enter.

> **NOTE:**
> The PROMPT command provides a way to change the DOS prompt that appears on the screen. The PROMPT command is described in Chapter 5, "Tips for Advanced Users" and in Chapter 6, "DOS Command Reference."

DOS Program Types

The DOS programs are of two basic types, internal and external.

Internal Programs

As soon as you start up your computer, the DOS internal programs are stored in the computer's memory. They are available at any time, just by entering the DOS command. For this reason, you won't find the internal DOS programs stored on the DOS disk.

The DOS internal programs are listed below.

Internal Directory Management Commands

CHDIR (CD)
DIR
MKDIR (MD)
PATH
RMDIR (RD)
TREE

Internal File Management Commands

COPY
DEL (ERASE)
RENAME (REN)
TYPE
VOL

Internal General Purpose Commands

BREAK
CLS
CTTY
DATE
EXIT
PROMPT
SET
TIME
VER
VERIFY

The uses of these commands are discussed in the DOS reference section in Chapter 6, "DOS Command Reference."

There are also some internal DOS batch processing commands. These commands are discussed in Chapter 4, "Using Batch Files."

Example: The DIR Command

An example of a commonly used internal DOS command is the DIR command (it stands for DIRECTORY). The DIR command tells the computer to display a complete list of the files stored on a disk.

Since the DIR command is used to activate an internal program, to call it into action you don't have to insert the DOS disk into the drive. Just type

dir

and press the **Enter** key.

DOS responds by immediately showing you a list of the files that are stored on the disk in the current drive. If the DOS program disk is in the current drive, DOS will display a list that looks similar to the list below.

```
Volume in drive A is DOS1
Directory of  A:\

COMMAND     COM    23612   11-17-87     12:00p
ANSI        SYS     1651   11-11-87     12:00p
APPEND      COM     1877   11-11-87     12:00p
ASSIGN      COM     1523   11-11-87     12:00p
ATTRIB      EXE     8234   11-11-87     12:00p
BACKUP      EXE    21720   11-11-87     12:00p
CHKDSK      EXE     9680   11-11-87     12:00p
COMP        COM     5770   11-11-87     12:00p
DISKCOMP    EXE     3808   11-11-87     12:00p
DISKCOPY    EXE     4096   11-11-87     12:00p
EDLIN       EXE     7356   11-11-87     12:00p
EXE2BIN     EXE     3050   11-11-87     12:00p
FASTOPEN    EXE     3919   11-11-87     12:00p
FC          EXE     2209   11-11-87     12:00p
FDISK       COM     5888   11-11-87     12:00p
FIND        EXE     6403   11-11-87     12:00p
FORMAT      EXE    11005   11-11-87     12:00p
GRAFTABL    EXE     8210   11-11-87     12:00p
GRAPHICS    COM     2298   11-11-87     12:00p
JOIN        EXE     9012   11-11-87     12:00p
KEYB        COM     3762   11-11-87     12:00p
LABEL       EXE     2750   11-11-87     12:00p
MODE        COM     3488   11-21-87     12:00p
MORE        COM      266   11-11-87     12:00p
NLSFUNC     EXE     3060   11-11-87     12:00p
PRINT       EXE     7464   11-11-87     12:00p
RECOVER     EXE     4129   11-11-87     12:00p
REPLACE     EXE     4852   11-11-87     12:00p
RESTORE     EXE    20156   11-11-87     12:00p
SHARE       EXE     8144   11-11-87     12:00p
SORT        EXE     1898   11-11-87     12:00p
SUBST       EXE     9898   11-11-87     12:00p
SYS         COM     4607   11-11-87     12:00p
TREE        EXE     8588   11-11-87     12:00p
XCOPY       EXE    11247   11-11-87     12:00p

36 File(s)   5744 bytes free
```

The DIR command contains instructions to first determine what files are stored on the disk and then display a list of those files on the screen.

DOS provides the DIR program as an internal program (always available in memory), because you will use it often.

Notice that in the list of DOS files above, DIR and the other internal DOS programs are not listed. That is the key to understanding the nature of the internal programs: they are not stored on disk, but are kept active in the computer's memory.

External Programs

External programs, unlike the internal programs, are not kept active in the computer's memory. In general, these programs are less commonly used than the internal DOS programs. They are stored on disk and if you look at the list of programs stored on the DOS program disk above, you can tell which programs are external just by the fact that they are listed. These programs are only available when you access them from a disk.

> **NOTE:**
> There may be some external operating programs listed on your DOS program disk that are not described in this book. Computer manufacturers sometimes create special operating programs that are needed when using their computers.

Example: The CHKDSK Command

An example of the external type of DOS program is the CHKDSK program. It is activated by the CHKDSK command.

Whereas the DIR command tells DOS to display a list of the files stored on the disk, the CHKDSK command provides a different type of information about stored files.

If you have the DOS diskette in drive A and you are working in drive A (the A prompt appears at the left side of the screen), you can use the CHKDSK command. Type in

chkdsk

and press the **Enter** key.

DOS will display information about the types of files stored on the disk. For example, a CHKDSK of the DOS diskette might look like this:

```
362496    bytes total disk space
41984     bytes in 2 hidden files
89792     bytes in 36 user files
5744      bytes available on disk

524288    bytes total memory
372592    bytes free
```

REMEMBER

An INTERNAL DOS program can be used at any time. An EXTERNAL DOS program can only be used when that particular DOS program is available on the disk in the drive and subdirectory you are currently using (subdirectories are explained in Chapter 2).

External DOS programs will have either a .COM or .EXE extension to the program name. Only filenames that have one or the other of these filename extensions are stand-alone operating programs. Filenames with other extensions are either data files or have special operating functions related to other programs. For example, using the DIR command, we might find these external OPERATING PROGRAMS stored on a disk:

```
CHKDSK     EXE     9680    11-14-87    9:01p
COMP       COM     5770    11-21-87    9:20a
FIND       EXE     6403    11-14-87    8:29a
FORMAT     EXE     11005   11-14-87    8:13a
```

Again using the DIR command, we might find these DATA FILES (created using a word processor) stored on a disk:

TEST	**TXT**	**1151**	**10-12-87**	**1:11p**
LETTER	**A01**	**1023**	**10-22-87**	**2:22p**
FILE-2		**1634**	**10-24-87**	**8:22a**
TT8-12	**DOC**	**5690**	**11-26-87**	**9:01p**

DOS Format Notation

The CHKDSK command described in the last section is actually the name of a program that is stored on the DOS disk. You activate that program by typing in the name of the program (CHKDSK) and pressing the **Enter** key. To get a DOS program to operate, you always type in its name and then press the **Enter** key.

But remember, to get the program to do what you want it to do, you have to type in the name (and any additional information that the program needs) using EXACTLY the right format notation. There is probably nothing more troublesome about using DOS than figuring out exactly how to enter all the proper DOS command options using the correct notation.

Some DOS programs are initiated with very simple commands. The example of the internal DOS command we looked at was the DIR command. It's very simple to use: you just type in the letters "dir" and press the **Enter** key. DOS responds by showing you a list of the files stored on the currently active disk. But, like many other DOS programs, the DIR command can be typed in with some additional options that make the command more useful.

The DIR format notation (the format notations for all commands are listed in Chapter 6) looks like this:

DIR [d:][path][filename][/P][/W]

While this DOS command format may look very complicated, it really isn't once you understand what it's trying to tell you. One of the things that makes the command format look complicated is the use of many square brackets to indicate parts of the command that are optional.

If you look closely at the DIR command format, you will notice that everything except the actual "DIR" name is enclosed in these square brackets. That means that everything except the "DIR" name is optional. You can type in "DIR" alone and the command will work just fine. This is true for quite a few of the DOS commands: many of them do something useful just by typing in the DOS command name.

But you should learn to use the options also. Learning to add the options after the command name makes the DOS command much more powerful.

The next section summarizes the use of the DOS notation system.

Notation Summary

The notation conventions described in this section are common to most DOS reference manuals. However, different forms of this notation may be used in other books; for example, other types of brackets may be used to indicate command options.

> NOTE:
> Certain characters are never used to indicate DOS command format, but are special characters you sometimes have to enter when using the command. These are part of the command "syntax." If one of these characters appears in the command's descriptive notation, it can be entered as part of the command. The special characters are:
>
> | | | |
> |---|---|---|
> | **space** | **comma (,)** | **colon (:)** |
> | **semicolon (;)** | **equal sign (=)** | **backslash (\)** |
> | **forward slash (/)** | **period (.)** | |
>
> The placement of these command syntax characters in a DOS command can be very important. Incorrect use of any one of these characters will often make the command fail to operate (or, even worse, fail to operate correctly).

The command syntax symbols are used in a standard way with the various versions of DOS implemented on different brands of computers: they will be used in the same way, no matter what type of computer you are using (as long as the computer is using a version of MS-DOS or PC-DOS).

A typical command format is

(commandname) [d:][path]filename [/OPTION]

To use this type of command, you would enter:

1. the command name
2. a space after the command name
3. a drive letter indicating the drive the command is to act on
4. a colon
5. a path designation (where the file is located on the disk)
6. a filename to act on
7. command options (if any)

The first part of the format notation is

(commandname) [d:]

This indicates that you type in the name of the command, a space, and then a drive designation letter. You do not type in the brackets; they just show that the drive letter is optional for this command. Also, you do not type in the "d:" that is inside the brackets. Instead, you type in the letter of the drive you want the command to act on (usually the location of the file you want the command to act on).

The next part of the format notation is

[path]filename

You would not enter the words "path" or "filename." Instead, you would replace "path" with the appropriate disk and directory information (where to find the file to act on) and replace "filename" with the name of a file that is stored on the disk.

NOTE:

Don't worry about what the "path" designation means right now. A path refers to locations on a disk; it is usually used with fixed disks. Path designations are discussed fully in Chapter 3, "Using Fixed Disks." The examples used in this chapter (and in Chapter 2) are generally for use with diskette drive systems. The examples will refer to drive A or drive B. Chapters 3 through 6 will provide information and examples for fixed disk systems.

Other options are often indicated in the format notation. These options usually modify the effect of the command and are usually indicated by a forward slash and then a letter or a number. The DIR command provides two options:

[/P] [/W]

You enter these options by typing the forward slash and then the letter. Again, you do not enter the square brackets: these are not part of the command syntax but are used to indicate that each of the letters refers to an option. The command will still work without using any of these options

DOS Notation Rules

The notation concepts can be summarized as a set of rules:

- The DOS command name is always entered first. It is usually written all in uppercase letters, but the command can be entered using either lower-case or uppercase.

- Always leave a space after the DOS command before entering any of the command options.

- When referring to a drive in a DOS command, you must always follow the drive designator with a colon (:). (This is how DOS recognizes it as a drive designation.)

  Entering the drive designation letter is often an option for a DOS command, but not always. Some commands are not related to disk drives and therefore do not require a drive designation. And when you are giving DOS commands that deal with disk drives you often will already be working in the drive in question; in that case, you do not have to enter the drive designation.

  For example, if you are working in drive A (the DOS prompt A> is showing at the left of the screen) and you want to use the DIR command to display a directory listing of drive A, you do not have to enter the drive designation. If you do not enter any drive designation, DOS always assumes you are referring to the drive you are currently working in (sometimes called the "default" drive).

- A pathname (path) refers to the path you want DOS to follow in order to act on the DOS command. It indicates the path from the current directory or subdirectory to the files that are to be acted upon.

- A filename (filename) is the name of a file stored on disk.

- Characters in a command format represented by a letter or a number and preceded by a forward slash (/A) are command options (sometimes known as "switches"). These options provide special functions that can be used with the DOS command. The command can always be used without these options, but use of the option expands the capabilities of the DOS function.

- Items enclosed in square brackets are optional; in other words, the command will work in its basic form without entering the information contained inside the brackets.

- Ellipses (. . .) indicate that an item in a command syntax can be repeated as many times as needed.

- When items are separated by a vertical bar (¦), it means that you can enter one of the separated items. For example:

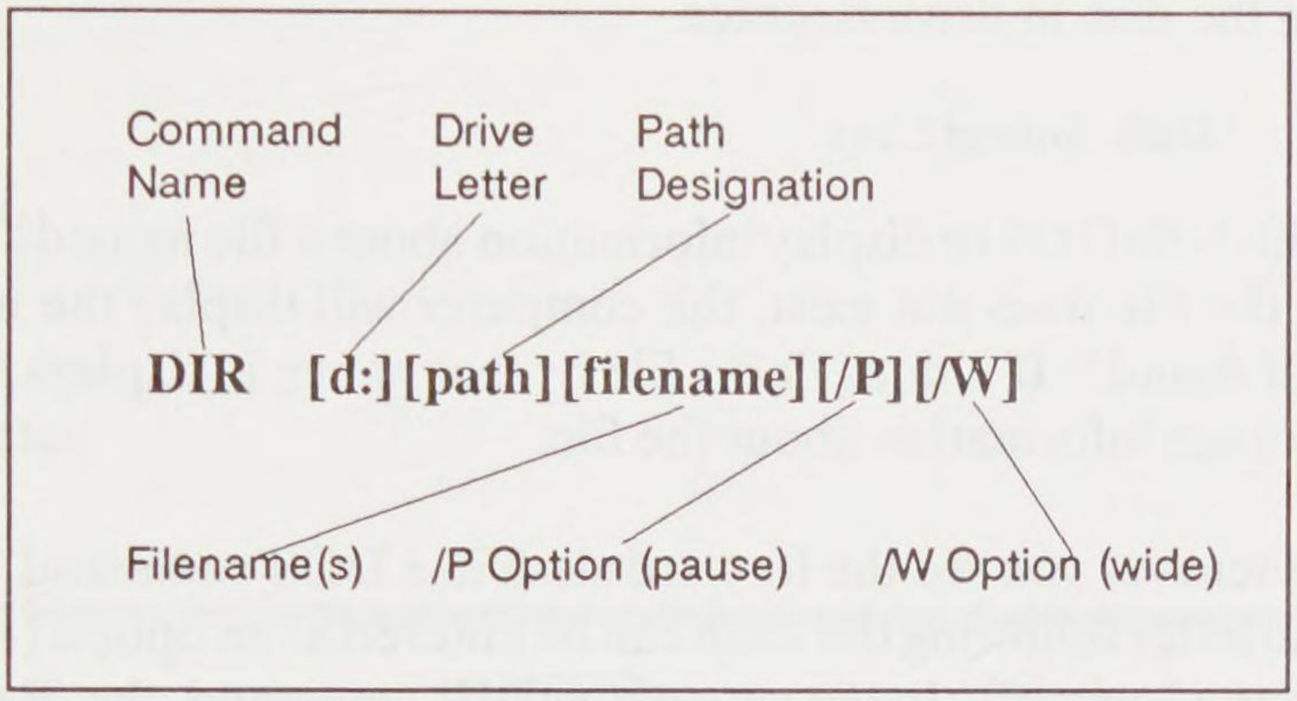
  ON ¦ OFF

  This means that you can enter either ON or OFF, but not both.

Notation Example: The DIR Command

You will find it easier to understand the complexities of the DOS command syntax notation if you look at an example. In this section, we will use the DIR command to look at each of the components of a typical DOS command syntax.

The illustration below shows the components of the DIR command.



Each of the elements of the illustrated DIR command are described below.

[d:] The small letter "d" is used to indicate "drive." The "d:" is a command option that is used with most of the DOS commands. The square brackets tell you that the drive designator they enclose is optional: the command will work without it (by acting on the currently active drive).

In other words, this option tells you that you can use the DOS command on any drive you specify. As indicated, you must follow the drive specification with a colon (to tell DOS that the letter is a drive designator). Do not leave a space between the drive indicator letter and the colon.

[path] This option is a feature common to most of the DOS commands. It is used to indicate a directory and subdirectory path to the location of the files to be acted on. (More information on how to use subdirectory pathnames is provided in the Chapter 3, "Using Fixed Disks.")

[filename] Many DOS commands can be used with a filename. A DOS filename is the name of a file that is stored (or is about to be stored) on disk. Filenames may also have a one to three letter extension following the main filename. Enter a period after the main filename and then, with no spaces, enter the extension. More information about using filenames is provided later in this chapter.

You can use the DIR command to determine if a particular file is stored on a disk. For example, to determine if the file TEST1.TXT is stored on the disk in drive B, enter

DIR b:test1.txt

This tells DOS to display information about a file named "TEST1.TXT" If the file does not exist, the computer will display the message, "File not found." If it does find a file by that name, it displays the usual disk storage information about the file.

[/P] [/W] Whenever you see the forward slash in a DOS command, it means that the letter following the slash can be entered as an option (often referred to as a "switch"). In the case of the DIR command, the /P option causes the screen display to pause when the screen is full. The /W option causes the display to list only the filenames, in a five-column display format, without the information about disk storage. Many more filenames can therefore be displayed on one screen.

There are a great variety of DOS command options. Each DOS command has its own particular options and the same option letter can mean something different for different DOS commands. Chapter 6, "DOS Command Reference" describes each of the command options in detail.

DOS Filenames

In order for DOS to properly handle file storage, you must name your files according to DOS rules. The basic rules are described below.

Filenames can include one to eight characters. The filename can be made up of letters, numbers, characters, or a combination of such characters. The filename can also include a one to three character extension. The characters that can be legally used for a DOS filename or a filename extension are:

The letters **A-Z.**
The numbers **0-9.**
These special characters: **~ ‘ ! @ # $ % & ( ) - _ { } ’**

All of the other characters on a standard PC keyboard are reserved for special purposes and cannot be used in a DOS filename.

The following are examples of valid DOS filenames:

Test1
TEST1
test1.txt
tests$-#.7
tester.@12
%.DOC
JUNE.(2)
TeSt1_#.Doc

> **NOTE:**
> As indicated in the first example filename "Test1" above, DOS recognizes both upper and lower case filenames. However, DOS converts all filenames to upper case; therefore, "Test1" and "TEST1" would be treated as the same file.

The following filenames won't work because spaces have been left within the name

TEST 1
TEST 2.DOC
MARCH.FI 1
TEST. 8

You can't use reserved characters in a filename. The reserved characters are listed below:

*	Asterisk	/	Slash
+	Plus sign	?	Question Mark
:	Colon	=	Equal sign
;	Semicolon	[ ]	Brackets
"	Quotation marks	\	Backslash
,	Comma	|	Column separator
.	Period	< >	Angle brackets

The following filenames won't work because the names include reserved characters.

TEST:1
test>2
TEST/1
test=doc
TEST[1]

Certain names cannot be used because they are used by DOS to refer to devices. You can't use the names:

AUX
COM1, COM2, etc.
C ON
LST
PRN
LPT1, LPT2, etc.
NUL

The following filenames or the filename extension includes too many characters. DOS will recognize only an eight character filename with a three character filename extension.

testfile1.doc
file12-11-87
letter.carl
test.11-2

WARNING:
If you use more than eight characters in a filename, DOS ignores the extra characters.

Since DOS will only act on the first eight characters of a filename, a file named, "TESTS1-10" is treated as if it was named "TESTS1-1." TESTS1-10 is treated as exactly the same file as TESTS1-11 (they are both seen by DOS as TESTS1-1). Because of this, you could lose the data in one of the files by mistakenly thinking you were storing two different files.

Designing Useful Filenames

When you first begin storing files on disks, you will probably give your files names that remind you of what is stored in the file. The problem is that a name which helps to remind you today, might appear to be a complete mystery in few months.

The fact that DOS restricts you to only eight characters can make creating useful names very difficult. DOS does allow you a three character extension to the filename, but the value of this extension may also be lost because some commercial programs automatically apply their own extension to any file. Typical examples are word processing programs that add a TXT or DOC extension to the end of all filenames. (Sometimes you can override this feature by entering your own filename extension.) Also, some programs automatically store a backup copy of the file you are working with and add a BAK extension to the filename, overwriting your filename extension.

The solution is to think up a standardized scheme for naming your files and stick to it. A common one is to use the first letter or two of the filename to indicate the type of file (l for letter, m for memo, a for appointments list, etc.) Then, you can use the remaining seven characters to provide information about what is in the file. For example, the filename, "LJJ4-18" could remind you that the file contains a letter to John Jones, written on April 18. As we will see in the next section ("Using Wildcards"), there are also other advantages to using a consistent pattern of naming files.

Using Wildcards

DOS "wildcards" are special characters you can use in conjunction with a DOS filename when you are specifying more than one file in a DOS command. DOS provides two different wildcards, the question mark (?) and the asterisk (*). Either can be used in a DOS filename instead of actual filename characters. The uses of these two wildcards is described below.

Using the Question Mark in Filenames

The question mark (?) is used to match a single character at a specific point in the filename. For example, in the filenames below, the only difference is the number.

TEST1
TEST2
TEST3
TEST4
TEST5

Therefore, you can use the question mark in any DOS command if you want to refer to all five of these files. For example, if you wanted to use the DIR command to determine how many TEST files were on the currently active disk you could enter

dir test?

DOS would respond by showing you all five of the above named files. Remember, however, that the question mark only replaces one character in the filename. If there was also a file named TEST10 on the disk, it would not be displayed by using the above version of the DIR command. However, you could get a directory listing of both the five TEST files and the TEST10 file by entering

dir test??

DOS responds to the above version of the DIR command by displaying all files that begin with TEST and end in any one OR two other characters. Be aware also that if DOS finds a filename that matches the characters OTHER THAN the question mark(s), that file will still be acted on. For example, if there was a file named "TEST" on the disk, it would be displayed by using the "dir test?" command.

You can use up to eight question marks in a DOS command that refers to a filename. You can also use up to three question marks to refer to the filename extension. For example, you could use the following version of the DIR command to get a directory listing of all the files on a disk that start with the letter "M":

dir m???????.???

But there is an easier way to do this as we shall see in the next section.

Using the Asterisk in Filenames

The asterisk (*) is used to refer to an unknown number of characters at the end of a DOS filename. In the above example, we used

dir m???????.???

to get a directory listing of all the files on a disk that start with the letter "M":

While that method works just fine, there is no need to type in all those question marks. The asterisk does the same thing. We can duplicate the above version of the DIR command by entering

dir m*.*

The asterisk takes the place of any number of characters in the filename following the letter "M." Notice that the asterisk also serves the same purpose in the filename extension. Two asterisks separated by a period (*.*) can be used to refer to all the files on a disk.

Some other examples of using wildcards can be found under the discussion of the COPY command in the next chapter.

Disks and File Storage

One of the main uses of DOS is to manage disk files. Many commercial programs provide you with a way to create files and store them on disk. Almost all of these programs store information on disks using standard DOS procedures and it's important that you know something about how disk storage is accomplished.

Although a variety of storage devices are available for DOS-based computers, most of today's personal computers use some type of diskette (often referred to as a "floppy disk").

Many computers also store and retrieve data using a fixed disk (sometimes known as a "hard disk"). MS-DOS versions 2.0 and later provide added support programs for using fixed disks. DOS treats both diskettes and fixed disks the same. DOS stores data on both types of mediums in the same way, using the same storage allocation system and using the same types of filenames.

Diskette Cautions

Use a little caution when handling the thin 5 1/4 inch "floppy" diskettes. They can be easily bent or marred. Insert diskettes into the drive with the label facing up and the notch on the left. The part with the oblong hole in the plastic outer case goes in first.

When inserting diskettes into the drive, don't use excessive force. They should slide in easily. Make sure the diskette drive door is unlocked and make sure there is not already a diskette in the drive. There are several types of disk drives, but each has a "locked" and "unlocked" position. With diskette drives there will be a slight resistance as the diskette is almost in and then a definite "feel" of solid resistance. If the diskette does not go in smoothly, remove it carefully and try again. After the diskette has been inserted, lock the drive latch. But never force the latch; if it does not close smoothly, remove the diskette and try again.

Label disks as you use them using a felt tip pen. Never write on the label with a pencil or ball point pen; you could scar the surface of the recording surface inside.

Be careful never to bend floppy diskettes. Never leave them in the sun or near other heat sources.

Diskettes are generally not as sensitive to cold or to everyday electrical and magnetic sources as some might think, but diskettes may be harmed in this way. Probably the most common cause of damage to diskettes is food and liquids. Keep your diskettes in their paper case and keep food and drinks away.

Disk Types

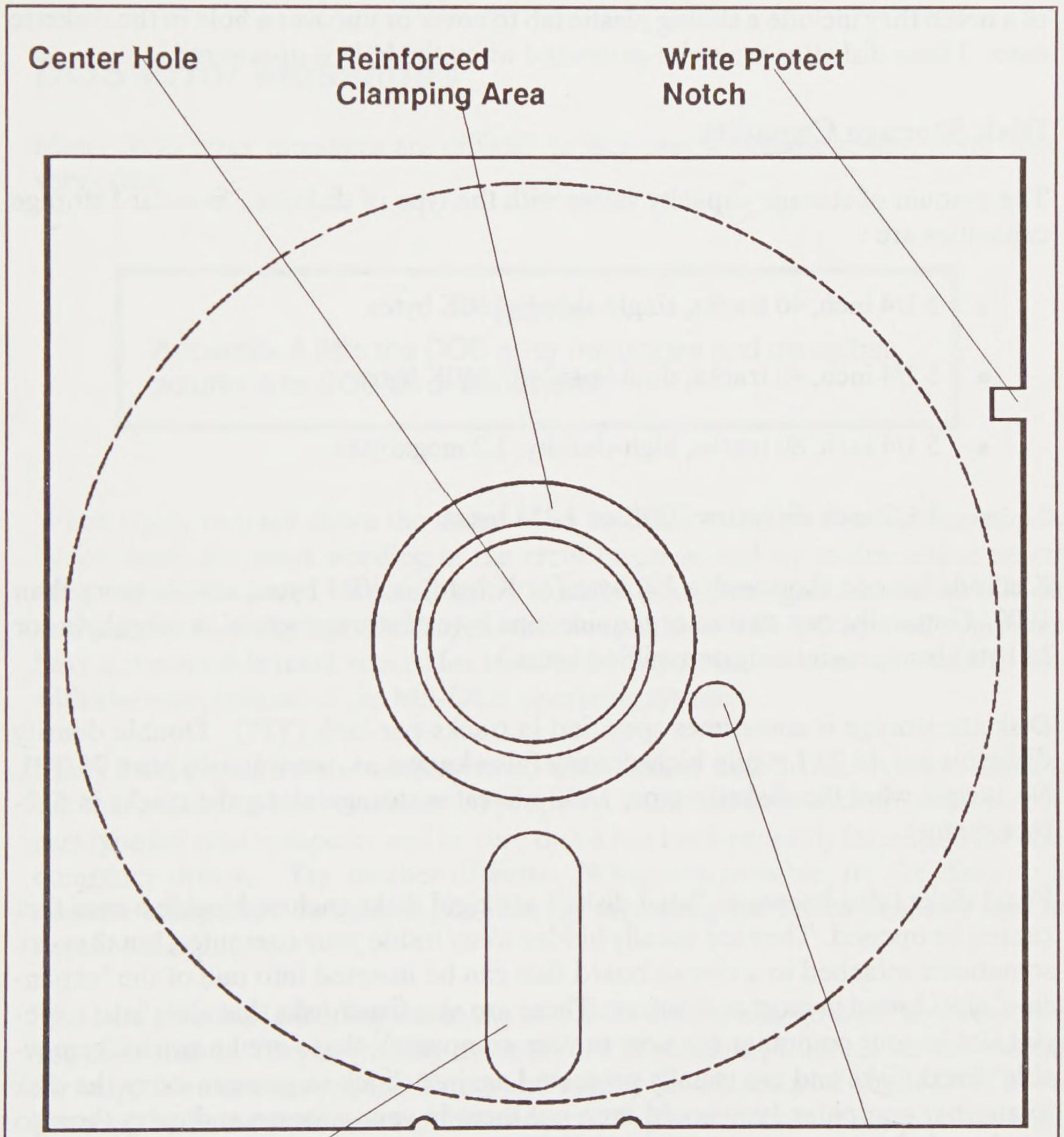
Disks are produced in a "raw" or unformatted state, but DOS provides a special program called FORMAT that prepares the disk to accept files for storage. The FORMAT program rearranges the magnetic patterns on the disk into a meaningful (to DOS) arrangement. When using DOS, the disk storage space is arranged into tracks (concentric circles from the center of the disk outward) and sectors (sections of each of these tracks).

Although PCs use several different data storage layouts on diskettess, the storage medium is always a circular platter of flexible plastic, thinly coated with the same type of magnetic brown-colored oxide found on other types of recording media such as sound recording tape.

A fixed disk uses the same type of magnetic coating, but the disks (a fixed disk usually has two or more platters) are rigid and are tightly sealed inside a metal container.

On diskettes, the thin oxide-coated platter is contained inside a protective plastic jacket. The thin 5 1/4 inch diameter diskettes that have been available for personal computers for several years are housed inside a flexible jacket.

If the outer casing from one of these diskettes was transparent, it would look like this:



The newer 3 1/2 inch diameter diskettes are sealed inside a more rigid jacket. The 5 1/4 inch floppy diskettes usually have a notch along the edge. If you cover this notch, a properly functioning diskette drive cannot write anything on or erase anything from the diskette. (Most boxes of new diskettes contain sticky-backed tabs that can be used to cover this notch.) This notch has become known as the "write-protection" notch. It is best to cover it on any diskette that holds important data. The newer 3 1/2 inch diskettes utilize a similar write-protection scheme: but instead of a notch they include a sliding plastic tab to cover or uncover a hole in the diskette case. These diskettes are write-protected when the hole is uncovered.

Disk Storage Capacity

The amount of storage capacity varies with the type of diskette. Standard storage capacities are :

- 5 1/4 inch, 40 tracks, single-sided: 160K bytes
- 5 1/4 inch, 40 tracks, double-sided: 360K bytes
- 5 1/4 inch, 80 tracks, high-density: 1.2 megabytes
- 3 1/2 inch diskettes: 720K or 1.4M bytes

K stands for one thousand; a kilobyte (or K byte) is 1024 bytes, a little more than 1000. Generally, one character requires one byte of storage space A megabyte (or M byte) is approximately one million bytes.)

Diskette storage is sometimes specified in tracks-per-inch (TPI). Double density diskettes are 48 TPI, while high-density (also known as quad-density) are 96 TPI. No matter what the diskette type, DOS allocates storage along the tracks in 512-byte sectors.

Fixed disks (also known as "hard disks") are rigid disks enclosed inside a case that cannot be opened. They are usually hidden away inside your computer, but they are sometimes attached to a circuit board that can be inserted into one of the "expansion" slots found in most computers. There are also fixed disks that slide into a special slot in your computer (or next to your computer); these are known as "removable" fixed disks and are usually protected against shock so you can carry the disk to another computer (you could even put them in your suitcase and carry them to another city).

Fixed disks are usually described in terms of their total formatted capacity. Because they hold so much data, the storage amounts are usually stated in megabytes, or millions of bytes. The smallest fixed disks usually used with PCs hold about 5 megabytes of data, the largest hold hundreds of megabytes.

Chapter 3, "Using Fixed Disks," provides much more information about the special DOS commands that make using fixed disks easier.

DOS Error Messages

Many DOS error messages are difficult to interpret; fortunately, they don't occur very often.

> **NOTE:**
> Appendix A lists the DOS error messages and describes solutions to DOS error conditions.

When trying to track down the cause of an error message, proceed systematically. Write down the exact wording of the error message and try to determine exactly what caused the error. If possible, try to make it happen again. Look for obvious and simple causes before deciding on drastic solutions. MS-DOS based computers have a remarkable track record for trouble-free operation -- once you learn to deal with the complexities of the MS-DOS operating system.

Many DOS errors have something to do with disks or disk drives. Always check to see that a diskette has been inserted properly. Make sure the diskette is of the correct type for your computer and be sure that it has been properly formatted for your computer drives. Try another diskette. Whenever possible, try the diskette in another computer. In fact, if possible, try repeating the entire problem task on another computer.

The best solution to finding the cause of a DOS error message is to "try everything you can think of." Try narrowing down the list of possible problems before you call for help. You can save yourself time and money (and learn a lot about DOS in the process) by being your own trouble-shooter. Try to eliminate every possible combination of possibilities before calling for help. You'll be surprised at how much you can learn on your own.

Chapter 2
Using Common DOS Commands

Of the 63 DOS commands, only a few are used on an everyday basis. This chapter describes a set of eleven commands that we have found to be widely used. If there is an "average" DOS user, these will generally be the commands that such a user will learn. You can carry out a great many useful DOS tasks using only these few commands; in fact, we have found that the majority of users manage to accomplish all of their file and disk management tasks using only the commands described in this chapter.

The best way to learn how to use DOS commands is to learn the things that the commands have in common. Chapter one presented an introduction to the "command format" using shorthand symbols that act as a guide to using the DQS commands. This chapter continues that introduction, using the most commonly used commands.

It is best to learn these frequently used commands before moving on to the less frequently used (and often more complex) ones. If you have problems with a command, the best solution is usually to enter one or more of the examples that are provided with the discussion of each command. Then, after you have seen how the command works, think through your own common DOS tasks. Determine how you could use these commands to make your tasks easier and try them out using your own data.

You can place your most frequently used command sequences in a "batch file." Batch files are used to carry out DOS commands (or to run other programs) in a predetermined sequence. For more information on using batch files, refer to Chapter 4, "Using Batch Files."

The real value of the commands presented in this chapter is to help you make better use of your computer's capabilities; and the reason these commands are more commonly used than the others is because they are very useful. If you get to know each of these commands well, you will find yourself using them easily and constantly. You will soon find that they make using your computer much easier.

Frequently Used DOS Commands

In this section, we take a closer look at the most commonly used DOS commands. Our experience indicates that the following DOS commands are among the most frequently used with computer systems that do not include a fixed disk.

CHKDSK
COPY
DATE
DIR
DISKCOPY
ERASE (DEL)
FORMAT
PRINT
RENAME (REN)
TIME
TYPE

> **NOTE:**
> There are also other commands that will be commonly used if your computer includes a fixed disk. These commands are discussed in Chapter 3, "Using Fixed Disks."

In the following sections, each of these commands is described, in detail.

CHKDSK

Type: External

Command Syntax:

CHKDSK [d:][path][filename][/F][/V]

The CHKDSK command is used to provide a status report on:

1. The amount of available space on a disk (either floppy or fixed)
2. The amount of computer memory
3. Any errors found on the disk.
4. Any files broken into more than one piece for storage on the disk.

The CHKDSK command is a useful command that should be used often. (It can also be used to check a fixed disk.) CHKDSK is a very valuable diagnostic tool, used too seldom by most DOS users. Use the CHKDSK command after you have been using a disk for some time, or after copying files to a disk. Like many of the DOS commands, its real power lies in its less well-known attributes.

If you enter the CHKDSK command alone, DOS checks the status of the disk you are currently working in. It issues a report that looks something like this:

```
Volume ____ created Jan 1, 1987 2:32a

362496 bytes total disk space
46080 bytes in 3 hidden files
259072 bytes in 37 user files
57344 bytes available on disk

524288 bytes total memory
258128 bytes free
```

The first four lines of this display refer to the files stored on the disk you specified; they indicate the total disk space (the storage capacity of the disk), the number of hidden (system) files, the number of user files, and amount of storage space still remaining to be used.

If you had created directories on the disk, they would be reported on an additional line. The last two lines do not refer to the disk you specified; they indicate the amount of memory (RAM) in your system and how much of it remains available for use.

DOS reports this message if it finds no problems. If DOS finds lost portions of files, it reports an additional message like this:

```
n lost clusters found in n chains
Convert lost chains to files (Y/N)?
```

What this rather cryptic message means is that DOS has found data on the disk it can't account for. A number will be displayed in place of n, depending on what DOS finds. DOS reports this type of message when it can't find all the links between blocks of data in files that have somehow become fragmented. This can happen if you are copying a file and the power goes off (or for a variety of other reasons).

CHKDSK Command Options

The /F option tells DOS to convert lost data fragments into normal text files. If you specified the /F option as part of the CHKDSK command, DOS can convert each

lost fragment to a new file and store them in the root directory of the disk it was found on. The first new file will be named "FILE0000.CHK" and each succeeding file will be named with the next higher number. If you didn't add the /F option, no problem: just run CHKDSK again, adding the /F right after the drive and path designation. For example, if you enter

```
chkdsk a: /f
```

CHKDSK will convert lost segments to files. If, for example, three lost clusters are found, DOS will display

```
3 lost clusters found in 3 chains.
Convert lost chains to files (Y/N)?
```

If you respond to this prompt by pressing the letter **Y** key, three new files will be created on the root directory of the disk. They will be named:

```
FILE0000.CHK
FILE0001.CHK
FILE0002.CHK
```

Each of these files will contain a lost cluster. You can use your word processor to look at these files to determine which file the lost segment came from. If the lost segment came from a text file you may be able to re-merge it with the main file. However, upon inspection, you may find that the original file is intact. This may indicate that the lost segment resulted from file management tasks carried out by your word processor (or other type of program). In this case, you can simply delete the newly created segment file.

If the lost segment came from a program file, you probably will not be able to re-merge it. If possible, replace the file by re-copying from the original source. If you cannot replace the file, check the program file that is stored on disk carefully by running all of its features. It may still be operational.

If you specify a filename as part of the CHKDSK command, DOS reports if the file is stored on the disk in a non-contiguous sequence. That is, when the file was stored, DOS could not find a space on the disk large enough to store the entire file. It therefore stored part of the file in one place, and part in another.

For example, if the file "TEST1.DOC" was stored in 8 non-contiguous sections, and you enter the command like this:

```
chkdsk b:\files\test1.doc
```

DOS would report:

B:\FILES\TEST1.DOC
Contains 8 non-contiguous blocks.

> **NOTE**
> There is nothing wrong with having non-contiguous files on the disk; DOS can keep track of them and treat them as one file. However, if you end up with too many of these types of files stored on the disk, the time it takes to work with your files will increase. Non-contiguous files become more likely when you are constantly adding and deleting files to and from a disk. Use CHKDSK to determine if many files are non-contiguous. If so, you can format a fresh disk and copy all of your files over to it. This will put the files back in order and speed up your disk access.

If you use the /V option as part of the CHKDSK command, DOS displays a series of messages indicating its progress as it checks the disk.

Many users think that this is all CHKDSK does, but it actually does much more. It looks for many other types of file storage problems on the disk and reports an appropriate error message if it finds problems.

Some of the error messages that may be displayed while you are using CHKDSK are described in Appendix A, "DOS Error Messages."

COPY

Type: Internal

Command Syntax:

COPY [d:][path]filename [d:][path][filename][/A][/B][/V]

or

COPY [/A][/B][/V] [d:][path]filename

or

COPY [d:][path]filename+[d:][path]filename[...]

The COPY command may be the most used of all the DOS commands. In its simplest form, the command is used to copy a file from one disk to another and is relatively easy to use. However, like many of the other DOS commands, the COPY command includes many powerful options.

If you can sort through the complicated-looking command format, you can pick out only two aspects that are not optional. They are:

COPY filename

Add to this the destination for the copied file and you have the simplest (and most commonly used) form of the COPY command. The following would be a typical usage:

copy test1 b:

This form of the command copies the file TEST1 (it must be stored in the current directory of the disk you are using) to the disk in drive B. The first part of the command (in this case, the file TEST1) is always the SOURCE file. The second part (after a space) is the DESTINATION (in this case, to drive B using the same name).

The rest of the COPY command options are of three types.

Option 1. COPY using pathnames

You can copy files to specific directories on any drive. For example:

copy c:\files\test1.doc b:\letters

would copy the file TEST1.DOC from the FILES directory on drive C to the LETTERS directory on drive B. Using this form of the command, you can copy files from any directory on any disk to any other directory (on the same disk or another). (Chapter 3, "Using Fixed Disks" presents more information on using directories.)

Option 2. COPY using a new name

You can copy to the same disk or even the same directory (as long as you change the name). The example above copies the file using the same name. If you want to change the name as it is copied, just add a new name (TEST1.BAK in this example) after the second drive and path designation, like this:

copy c:\files\test1.doc b:\letters\test1.bak

Option 3. COPY combining files (append)

The previous examples represent the forms of the COPY command you will use most often. However, some of the less well-known COPY options are also very useful. For example, if you want to copy several files into one longer file, you could use a form of the command that adds the plus sign (+) between file specifications. For example:

```
copy  test1.doc+test2.doc  b:test.add
```

This form of the command combines the files TEST1.DOC and TEST2.DOC and stores them in a new file on the disk in drive B named "TEST.ADD." The files are combined in the order you entered them.

This form of the command is very useful for combining text files. Working with large files using a word processor can be troublesome; this form of the COPY command lets you work with smaller files (say, a chapter at a time) and then combine them into a larger document later.

A very useful form of this type of combined copying uses the DOS wildcards that were described in Chapter 1. For example:

```
copy  test*.*  b:tests.add
```

finds all files beginning with the letters "TEST" on the current drive and copies them all into one file titled "TESTS.ADD" on the disk in drive B. The files are copied in the order they appear in the directory. If you want to combine files into a specific order, use the plus sign version described above.

COPY Command Options

The COPY command provides three "switch" type options. They must be entered in the order indicated in the command syntax description above. The three options are:

/A When used prior to the SOURCE file specification, this option causes the files specified to be treated as ASCII (text-only) files. The text in the file is copied until DOS encounters an end-of-file marker (the Ctrl-Z character) in the file.

When used prior to the DESTINATION file specification, this option causes the file(s) to be copied as an ASCII (text-only) file and it adds an end-of-file marker (the Ctrl-Z character) to the end of the file.

/B When used prior to the SOURCE file specification, this option causes the files specified to be treated as BINARY files. They are copied in their entirety (end-of-file markers, if any, are ignored).

When used prior to the DESTINATION file specification, this option causes no end-of-file markers to be added to the end of the file.

/V This options tells DOS to check (verify) to be sure that the file that is being created exactly matches the original.

DATE

Type: Internal

Command Syntax:

DATE

DATE mm-dd-yy

This command gives you a way to display or set the date for the system. The date is used by the system to "date-stamp" any file you create or modify: whenever you create a file, the time and date currently set are stored along with the file name and the file size.

Unless you have created an AUTOEXEC.BAT file, each time DOS is loaded, the system asks for a date to be entered. If this request is ignored (you can press the **Enter** key to skip over it), the date stamp remains set at whatever date was previously in the system.

Since this is an internal command, you can enter "DATE" at any time (while working in any drive or directory) and the system will prompt you to change the existing date.

Some computers have a battery backup system that keeps track of the date even after the computer has been turned off. In that case, this command rarely needs to be used.

You can also use a somewhat complicated form of the PROMPT command to display the date (and time). Refer to Chapter 5, "Tips for Advanced Users" for more information.

DIR

Type: Internal

Command Syntax:

DIR [d:][path][filename][/P][/W]

As described in Chapter 1, the DIR command tells the computer to display a list of the files stored on disk.

The command can be used to display a list of the files on any disk (and in any sub-directory). It can be used with wildcards to display selected files. (Several examples are shown in the "Using Wildcards" section in Chapter 1.)

DIR Command Options

The command provides two options:

/P This option tells DOS to display only one screen full of data at a time. In case there are more files listed on a directory than will fit on the screen, this option pauses the scrolling of files when the screen is full. To restart the display of the list, press any key.

/W This option is also used to get a more readable display, but it does it by displaying less information about each of the listed files. This option causes a "wide" display, with only the filename and extension being shown in five columns.

Refer to Chapter 1 for examples of use of the DIR command.

DISKCOPY

Type: External

Command Syntax:

DISKCOPY [d:] [d:][/1]

This command is one of the most useful DOS commands. It should be used to make backup copies of all of your important diskette files.

The simplest version of this command is just the name of the program, "DISKCOPY." If you enter the name, DOS makes an exact copy of the SOURCE diskette (the diskette you insert first) on the DESTINATION diskette. This "name-only" form of the command tells DOS to do all of the copying in the same drive. It is the form of the command you will use if you only have one diskette drive.

Make sure the DOS program diskette is in the diskette drive and then enter:

diskcopy

and press the **Enter** key. DOS displays the following message:

Insert the SOURCE diskette in drive A:

Press any key when ready . . .

Remove the DOS diskette from the drive and insert the diskette you want to copy FROM (the SOURCE diskette) into the drive. Press any key to start the copying process. DOS tells you when it is time to remove the SOURCE diskette and insert the diskette you want to copy TO (the DESTINATION diskette).

If there are more than a few files on the disk, you will probably have to swap the diskettes back and forth a few times in order for DOS to get all the data transferred to the new diskette. Be very careful you don't mix up the two diskettes and start copying TO the original diskette. The way to be sure this can't happen is to write-protect the SOURCE diskette by putting a write-protect tab over the notch along the edge of a 5 1/4 inch floppy diskette (or opening the slide that covers the square hole in a 3 1/2 inch diskette). (Diskette types and write-protection were described in Chapter 1.)

Using DISKCOPY is much easier if you have a two-drive computer system. If you have both an A and B drive, enter:

diskcopy a: b:

and press the **Enter** key.

This form of the command tells DOS that you will put the SOURCE diskette in drive A and the DESTINATION disk in drive B. DOS will prompt you to do this by displaying:

Insert the SOURCE diskette in drive A:
Insert the DESTINATION diskette in drive B:
Strike any key when ready...

Put the diskettes in the drives and press any key to start the copying process. DOS takes care of the rest.

> **NOTE:**
> Some early versions of DISKCOPY do not format the DESTINATION diskette before it receives the copy, but all recent versions do.

The DISKCOPY program has only one switch-type option. It is the /1 switch which tells DOS to copy only the first side of a two-sided diskette.

DEL (ERASE)

Type: Internal

Command Syntax:

DEL (ERASE) [d:][path]filename

The DEL command (it can also be entered as "ERASE") is very useful, but it is also very dangerous. It is used to erase files from the disk. Since it is an internal command, it can be used at any time, while using any disk, in any subdirectory.

Using the ERASE command is very simple – almost too simple for the damage that can be done with it: just enter the name ERASE followed by the name of the file you want to erase from the disk. For example, to delete the file "SALES3.TXT" from the disk in drive B, enter

del b:sales.txt

The use of DOS wildcards makes this an even more powerful command. For example, you could erase all of the files on a disk that began with the letters "TEST" by entering

erase test*.*

You could erase all of the files in a subdirectory named "LETTERS" by first changing to the LETTERS subdirectory (refer to the CHDIR command in Chapter 2) and then entering

del *.*

This special use of wildcards with the ERASE command is so dangerous that before the files are erased, DOS displays the message

Are you sure (Y/N)?

If you press the letter **Y** key, all of the files in that subdirectory are immediately erased.

Recovering Deleted Files

There is nothing in the DOS documentation to indicate that once erased, a file can be recovered. However, there are several file recovery programs on the market that do just that. By using one of these programs you may be able to recover a file that has been accidentally erased.

All of these programs take advantage of a little known fact about how the DOS ERASE command works. To the user, an erased files appears to be completely gone. If you use the DIR command to see a list of all the files on the disk, there is no trace of an erased file. However, the fact is, DOS does not go to all the trouble of physically erasing files from the disk. Instead, it merely erases the filename from its own internal directory of files (the information is kept in a special place called the File Allocation Table, usually abbreviated as FAT).

When you erase a file, in effect, DOS tells its own record-keeping system to open up that part of the disk for new files. The truth is, the erased file is still occupying that space on the disk UNTIL YOU WRITE OVER IT with another file. So, as long as the original file area has not been written on, you can still recover the erased file. These recovery programs "talk you through" the process of finding the partially erased filename and then they attempt to recover as much of the file as is still intact by telling DOS to once again reserve that space on the disk.

FORMAT

Type: External

Command Syntax:

FORMAT [d:][/1][/2][/4][/8][/N:(sectors)][/T:(tracks)][/B][/S][/V]

This command is used to carry out a special DOS function, the preparing of disks to accept files. Before a diskette can be used by DOS, it must be formatted using the FORMAT command.

This command initializes the directory and file allocation tables (reserved areas on the disk used to keep track of files). As the disk is formatted it is analyzed for defects.

> **NOTE:**
> Like diskettes, fixed disks must also be formatted using the FORMAT command. However, since most fixed disks that are included in computer systems have already been formatted, you may never have to do it. If you do need to format a fixed disk, be sure to read your hardware user's guide first. Many manufacturers provide a special fixed-disk formatting program along with the usual DOS FORMAT program.

As with the ERASE command, you should use FORMAT with caution. Since all the data on the disk is erased by the formatting process (and it generally can't be recovered), you should be careful to specify the drive accurately.

The typical use of this command is very simple:

```
format  b:
```

This tells DOS to format the diskette in drive B. Before the formatting process starts, DOS displays:

```
Insert new disk in drive B:
and press any key when ready.
```

When you have inserted the diskette to be formatted in drive B (and you are sure you want to destroy all the existing data that is already on that disk), press a key to start the process.

The screen displays:

```
Formatting drive B: ....
Format Complete

362496 bytes total disk space
362496 bytes available on disk

Format another? (Y/N)
```

Press N (for NO) if you are finished formatting disks.

There are several options that can be used with the FORMAT command, depending on which version you are using. The early versions of DOS include very few FORMAT options because the early systems used only a few basic types of diskettes. But as a variety of new diskette types became available, later versions of DOS began to offer a new set of options, one for each type of diskette.

FORMAT Command Options

/1 This option formats a 5 1/4 inch diskette for single-sided use, regardless of the drive type. Using this option (along with the /8 option) makes your diskettes compatible with some of the earlier systems that used only one-sided disks, formatted for eight sectors per track.

/2 This option formats a 5 1/4 inch diskette for single-sided use, regardless of the drive type.

/4 This option provides a way to format a 5 1/4 inch double-sided, double-density diskette with a high-density (sometimes referred to as quad-density) drive.

> **NOTE:**
> This option is used with later DOS versions that support the high-density drives. However, the resultant diskette should not be used in any drive other than your high-density drive: our experience indicates that data written to such diskettes may not be reliably used with or copied to other computers.

/8 This option formats a 5 1/4 inch diskette for eight sectors per track. This setting is required if you are going to use diskettes with computers that use the earlier DOS versions. If you do not specify this option, the diskette is formatted at 9 or 15 sectors per track, depending upon the disk drive type. The /8 format option was used with DOS versions 1.x.

> **NOTE:**
> The /V parameter cannot be used with the /8 parameter.

/n:(sectors) Specifies the number of sectors per track on the disk. Used to format a 3.5 inch disk with the number of sectors per track specified using this option. For 720 K-byte disks, this value should be entered as "N:9."

/t:(tracks) Specifies the number of tracks on the disk. Used to format a 3.5 inch disk with the number of tracks specified using this option. For both 720 K-byte disks and 1.44 K-byte disks, this value should be entered as "T:80."

/B This option causes the FORMAT program to create an eight sector per track diskette with space allocated for the DOS system files, IO.SYS and MSDOS.SYS (IBM uses IBMBIO.COM and IBMDOS.COM). It does not transfer these two files or the COMMAND.COM file to the diskette, but if this option is used, these files can be transferred later using the DOS SYS command.

NOTE:
The /V and /S options cannot be used if you specify the /B option.

/V Causes FORMAT to prompt for a volume label after the disk is formatted. The label can be of 1 to 11 characters. The same characters acceptable in filenames are acceptable in the volume label.

NOTE:
The DOS LABEL command can be used to create, alter, or delete this type of volume label.

/S This option is used to transfer the DOS system files IO.SYS, MSDOS.SYS, and COMMAND.COM to the diskette as it is formatted. These system files are hidden files and will not appear in a directory listing. If this option is used, the resulting diskette is usable as a "boot" diskette; that is, it can be used to load the system files when DOS is first loaded. Use this option when you want to use the diskette as a "start-up diskette." Using some versions of DOS, /S must be the last option entered.

PRINT

Type: External

Command Syntax:

PRINT

PRINT [/B:buffersize][/D:device][/M:(maxtick)][/Q:(value)]
[/S:(timeslice)][/U:(busytick)][/C][/P][/T] [d:][path][filename] [...]

This command prints one or more stored text files on a printer. The files can contain any type of text, created by any program, as long as they are stored on the disk as standard ASCII characters.

You can use other DOS commands or other programs while the PRINT program is in operation. This is often referred to as "background" printing. You can "queue-up" files: enter as many files to print as you like, repeating the command format for each file (leave a space before each new format).

If you use the PRINT command without entering any options, DOS displays the current contents of the print queue. This does not change the status of the files currently in the PRINT queue. If you do not enter any of the forward slash options (/), the /P option is assumed.

A very effective way to use the PRINT command is by specifying groups of files using wildcards. For example, you could print all files in the current subdirectory that started with the characters "FIL" by entering

print fil*.*

Different computer manufacturers may implement different versions of the DOS PRINT program; therefore, there may be other options available with your version of PRINT.

NOTE:
Although there is a PATH option in the command format for the PRINT command, with some DOS versions, you can only print one file in another drive or subdirectory. In this case, if more than one file has been queued up for printing, DOS looks for these files only in the current drive and subdirectory. However, after you have entered the appropriate PRINT command format, you can change drives and/or subdirectories to use other programs. The printing of queued files continues without interruption.

If the files to be printed are on a diskette in the current drive, that disk must be left in the drive throughout the printing process.

/C — Provides a way to select which files to cancel. A filename already in the print queue that precedes this option will be suspended. Names of files (already in the print queue) that follow this option will be canceled. You can also list more files to print as part of the same PRINT command by entering the /P option followed by the filenames to print. For example, if you enter

print fil*.*/c

Any files currently in the print queue that start with the characters "FIL" will not be printed.

To cancel the printing of selected files, enter

print file1.txt/c notes.doc sales

This form of the PRINT command cancels the printing of the files "file1.txt," "notes.doc," and "sales."

/P — Sets the print mode. The preceding filename and all following filenames will be added to the print queue. You can also list files to cancel as part of the same PRINT command by entering the /C option followed by the filenames to cancel. For example, if you enter

print file1.txt/c file2.txt/p notes.doc

the file "file1.txt" is removed from the print queue and the files "file2.txt" and "notes.doc" are added.

/T This option forces DOS to terminate printing of previously queued files. When this option is entered while a print is under way, the printing stops and a cancellation message is printed. The paper advances to the next page. If you enter the name(s) of one or more new files to be printed at the same time you enter this option, the first file starts printing on the new page.
For example, if you enter

print /t

DOS immediately stops the printing and the print queue is emptied. Any additional command options are ignored.

The following options are allowed only the first time you run the PRINT command:

/B This parameter sets the size (in bytes) of the amount of memory used by the PRINT buffer. Increasing this value speeds up the operation of the PRINT command. If you do not use the /B option, the PRINT buffer will be set to 512 bytes.

/D:(device) Specifies the print device. If you do not specify this option, PRINT will prompt you to enter the name of a print device. The default is PRN.

/M:(maxtick) Specifies how many computer clock ticks PRINT can have to print a file. The value of /M can be from 1 to 255. If you do not use the /M option, this parameter will be set to 2.

/Q:(value) Specifies the maximum number of files that are allowed in the print queue (from 4 to 32). If you do not use the /Q option, this parameter will be set to 8.

/S:(timeslice) Specifies the time slice (the interval of time used by DOS to schedule PRINT jobs. If you do not use the /S option, this parameter will be set to 8.

/U:(busytick) Specifies the number of computer clock ticks that DOS will wait for a printer that is not available. If the wait is longer than this value, DOS gives up its time slice (refer to option /S below). If you do not use the /U option, this parameter will be set to 1.

Starting PRINT

The first time you enter the PRINT command (after the computer is started), DOS displays the following message (unless you specify the /D parameter):

Name of list device [PRN]:

Often, you can just press the **Enter** key to start the printing. This accepts the PRN option as the standard output port.

If this standard output port option doesn't work, you must enter the name of the port for the printer. The usual options are LPT#, PRN, COM#, or AUX. If you are using only one device hooked to a parallel port, you would usually enter LPT1. Using only one serial device, you would usually enter COM1. If these output ports are being used by another device (or if you have more than one printer attached to you computer), you may have to use LPT2 or COM2. If you have more than one other device, use the next higher number to specify the output port. If you are not sure, try experimenting. If you are experimenting, be sure to wait several minutes for the printing to start before you decide that its not working.

During use of the PRINT command, DOS may display one or more messages. The messages and their meanings are summarized below.

Message: List output is not assigned to a device

Explanation: The name of the printer you specified was not valid.

Message: PRINT queue is full

Explanation: You attempted to enter more than 10 files in the PRINT queue.

Message: PRINT queue is empty

Explanation: There are no files in the print queue.

Message: No files match d:(filename)

Explanation: No files match the filename specification you entered.

Message: All files canceled by operator

Explanation: You entered the /T (terminate) option to cancel all the files currently in the PRINT queue.

Message: File canceled by operator

Explanation: You entered the /C (cancel) option to cancel a the file currently in the PRINT queue.

Examples

To stop printing and cancel the print queue, enter

```
print /t
```

To remove the files "LETTER.TXT, "START.EXE," and "RATE.COM" (all on drive B) from the PRINT queue, enter

```
print b:letter.txt/c b:start.exe b:rate.com
```

To remove the file "SALES.DOC" from the PRINT queue and, at the same time, add the files "OLDSALE.DOC" and "NEWSALE.DOC," enter

```
print sales.doc/c oldsale.doc/p newsale.doc
```

RENAME (REN)

Type: Internal

Command Syntax:

```
RENAME (REN) [d:][path]filename filename
```

This command is used to change the name of a file stored on disk. The first filename specified is the name of the stored file. The second filename is the new name for that file. Notice that the drive and PATH specifications are options only for the first filename entered (the stored file to be renamed). Despite the name change, the file remains stored in the same location.

For example, if you enter

```
ren c:\files\sales.doc newsale.doc
```

the file "sales.doc" that is stored in the FILES subdirectory on drive C will be renamed "newsale.doc."

TIME

Type: Internal

Command Syntax:

TIME
TIME hh:mm[:ss][.cc]

This command gives you a way to set the time for the system. The time is used by the system to "time-stamp" any file you create or modify: whenever you create a file, the time and date currently set are stored along with the file name and the file size.

Unless you have created an AUTOEXEC.BAT file, each time DOS is loaded, the system asks for a time to be entered. If this request is ignored (you can press the **Enter** key to skip over it), the time stamp remains set at whatever time was previously in the system. Since this is an internal command, you can enter "TIME" while working in any drive or subdirectory and the system will let you change the existing time.

You can enter the TIME command alone and DOS will prompt you when to enter the time. You can also enter the time when you enter the TIME command. Either way, you must enter the time in the following format:

hh:mm:ss.cc

where **hh** is a one or two digit hours designation, **mm** is a one or two digit designation of the minutes after the hour, **ss** is a one or two digit designation of seconds after the minute, and **cc** is a one or two digit designation of the hundredths of a second after the second. The following numbers are allowed:

hh	0-23 for hours
mm	0-59 for minutes
ss	0-59 for seconds
cc	0-99 for hundredths of seconds

Only the hour and minutes settings are required. The following are allowable settings:

02:34
3:04
10:2:04
18:58:07.33
2:3:4.5

> **NOTE:**
> The version of DOS used on your computer may require slightly different forms of this time entry format.

Some computers have a battery backup system that keeps track of both the date and the time, even after the computer has been turned off. In that case, this command will rarely need to be used.

You can also use a somewhat complicated form of the PROMPT command to display the time (and date). Refer to Chapter 5, "Tips for Advanced Users" for more information.

TYPE

Type: Internal

Command Syntax:

TYPE [d:][path]filename

This command is used to display the contents of a file. If, for example, you wanted to view the TEST1.TXT file in your working subdirectory, you would enter

type test1.txt

and press the **Enter** key.

You can display the contents of a file in any subdirectory, on any disk, by adding the path designation.

The file is displayed exactly as it is stored on disk; therefore, if your word processor adds special formatting characters or if it stores files in a compacted form, this command will not be very useful. Nevertheless, most users use the TYPE command frequently — usually to quickly view a file to determine its contents.

The TYPE command displays the entire contents of the file. However, you can stop the scrolling data in either of two ways.

First, you can interrupt the data display by pressing and holding the Control key and then pressing the letter C key. This **Ctrl + C** key combination can be used to interrupt most programs.

Second, you can temporarily interrupt the display of data by pressing and holding the **Ctrl** key and then pressing the **Num Lock** key or the **S** key. This stops the data display until you press any key to restart it.

You can use the TYPE command to even greater benefit by using it to quickly send a "carbon copy" of a file's contents to your printer. There are two ways to do this using the TYPE command.

1. Printing using the Shift/PrtSc option

If you press and hold the **Shift** key and then press the **PrtSc** key, DOS sends an exact copy of whatever is appearing on the screen to the printer. You can use the TYPE command to get a quick print of any part of a stored file. First, use TYPE to display a file. When you have the portion of the file's contents on the screen that you want printed out, stop the display by using the **Ctrl** + **C** or **Ctrl** + **Num Lock** key combinations described above. Then, (after making sure your printer is turned on) hold down the **Shift** key and press the **PrtSc** key. Whatever is showing on the screen will be printed immediately.

2. Printing using the Ctrl/PrtSc option

The second method tells DOS to send the entire contents of the file to the printer. First, make sure the printer is turned on and then hold down the **Ctrl** key and press the **PrtSc** key once. After that, anything that appears on the screen is also printed.

Therefore, if you use the TYPE command in the normal manner, as the file is displayed on the screen it is also printed by the printer. You can stop the display (and the printing) at any time by pressing either the **Ctrl** + **C** or **Ctrl** + **Num Lock** key combination that was described above. This printing mode stays in effect until you press the **Ctrl** + **PrtSc** key combination again.

Other Useful Commands

Our experience indicates that these are the most frequently used of the DOS commands; however, there are a number of other DOS commands you will use regularly. The PATH command and the APPEND commands can be used to direct DOS to your most frequently used programs. The BACKUP, RESTORE, and XCOPY commands should be used regularly to make backup copies of your files and programs. A number of commands related to the use of fixed disks will be described in the next chapter.

The complete set of DOS commands is fully described in Chapter 6, "DOS Command Reference."

Chapter 3
Using Fixed Disks

A fixed disk (also known as a "hard disk") is a special type of disk storage device that can store very large amounts of data. As was described in Chapter 2, DOS stores data on a fixed disk using the the same format it uses for diskettes. However, because such large amounts of data can be stored on fixed disks, DOS versions 2.0 and later provide some special commands that, although they can be used with diskettes, are especially useful when dealing with the larger storage capacities of fixed disks.

These new commands provide a way to organize the data on a fixed disk into interrelated storage areas referred to as "directories" and "subdirectories."

Using DOS Directories

DOS directories and subdirectories are organized into branching storage areas that can be thought of as something like an upside down tree. The system is referred to as a "tree-structured" directory system.

When the fixed disk is first formatted, only one directory is created. This directory is referred to as the "root" directory and all other directories should be thought of as falling "within" or "under" the root directory.

> **NOTE:**
> Strictly speaking, any directory you create on a disk is actually a subdirectory within or "under" the root directory. However, in this book, we will often refer to both directories and subdirectories as a directory.

DOS provides a special command, MKDIR, to create and name the new directories. The rules for naming directories are the same as for naming files (one to eight characters followed by an optional extension of a period and one to three characters). As an example, let's say you have just formatted a disk (it can be a fixed disk

or a diskette) and you want to create a special directory called "DOS" to store all the DOS programs in.

If you are working in drive C (typically used to designate a fixed disk drive), you can create and name the new directory by entering

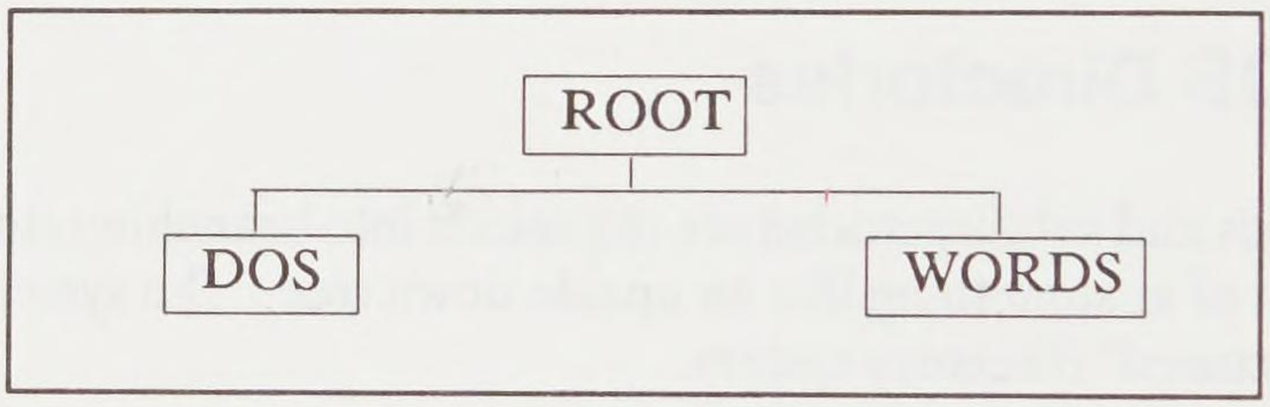
mkdir dos

The new directory named "DOS" will be established. Since no special reference to a drive or another directory was made, the DOS directory will be created within the current directory, whatever it is.

Let's also create another directory, "WORDS" that we might use to store text files. Enter

mkdir words

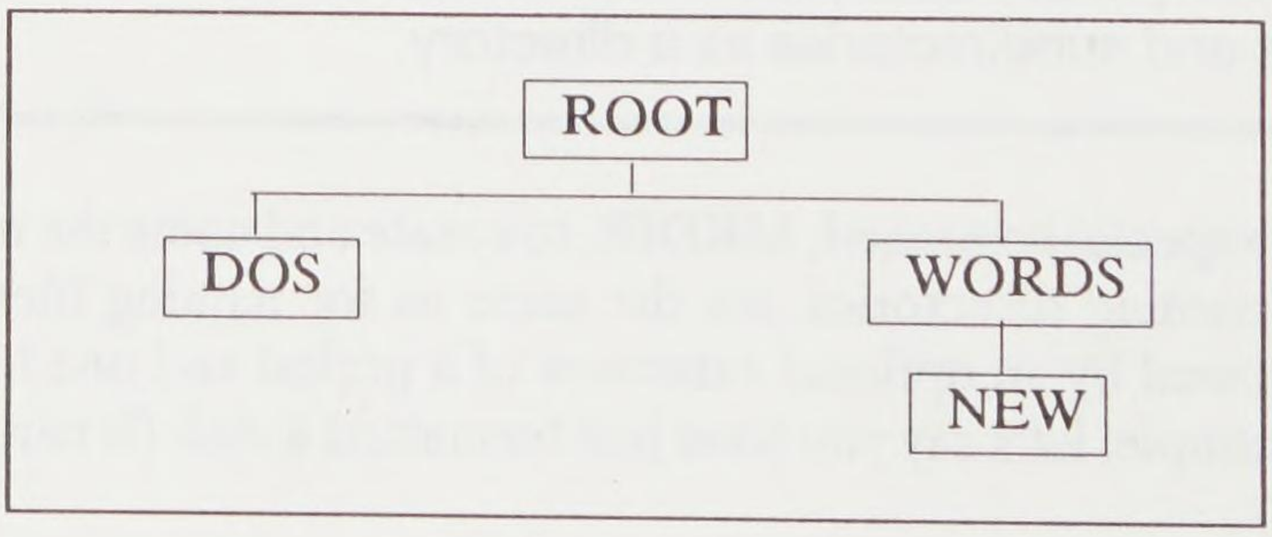
The resulting tree-structured directories can be represented with the drawing below.



We could also create more directories within these two directories. For example, if you were working in the "WORDS" directory and you entered

mkdir new

A directory labeled "NEW" will be created within the "WORDS" directory. It can be represented like this:



You do not have to be working in a directory in order to create a new directory within it. We could have created the directory described in the previous example from any drive or directory by entering

```
mkdir  c:\words\new
```

This form of the MKDIR command creates the NEW directory within the WORDS directory which is itself within the root directory. The MKDIR command is described in detail later in this chapter.

DOS provides another command, the TREE command to display the current status of your directories. For example, if you were working in drive A with the DOS diskette in that drive (the TREE program is stored on the DOS diskette), you could enter

```
tree  c:
```

to see a display of the directories on drive C. The resulting screen display (using the directories created in the above examples) should look something like this:

```
DIRECTORY PATH DISPLAY FOR VOLUME ___

Path is: C:

Subdirectories:

DOS
WORDS

Path is C:\DOS

No subdirectories found

Path is C:\WORDS

Subdirectories:

NEW
```

As indicated by this display, the "DOS" and "WORDS" directories are within the root directory and the "NEW" directory is within the "WORDS" directory. Notice especially the phrase "Path is C:\DOS." DOS uses the backslash (\) to indicate the root directory. When you use DOS commands, you will often need to make reference to directories by using this type of pathname (if you don't use a pathname, DOS assumes you are referring to the current directory). The pathname in the

example uses the "\DOS" notation to indicate that the DOS directory is found within the ROOT directory (as indicated by the backslash).

More information on using pathnames can be found in a later section of this chapter titled, "Using DOS Pathnames."

Special Fixed Disk Usage Commands

DOS provides six commands related to fixed-disk usage. They are:

Command	Shorthand	Type	Purpose
MKDIR	MD	Internal	Make a directory
RMDIR	RD	Internal	Remove a directory
CHDIR	CD	Internal	Change directory
PATH	none	Internal	Search directories for programs
APPEND	none	External	Search directories for files
TREE	none	External	Display directory paths

These commands all have to do with directories. Knowledge of these directory-related DOS commands is very important when using a fixed disk.

The following sections provide more complete explanations of each of these commands.

MKDIR (MD)

Type: Internal

Command Syntax:

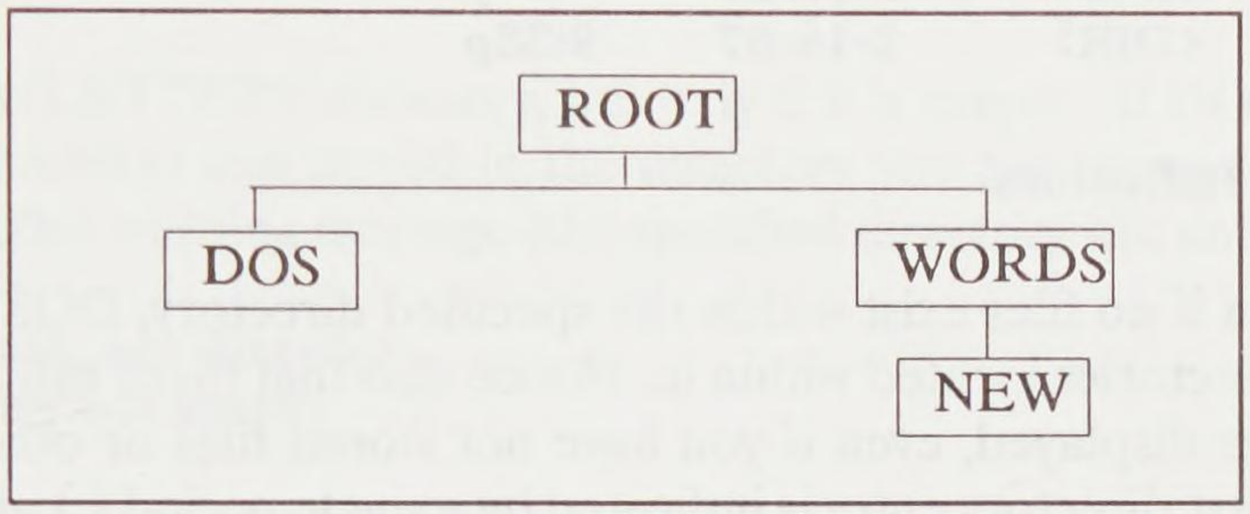
MKDIR (MD) [d:]path

The MKDIR command is used to create and name a new directory. The new directory can be within the root directory or any existing directory.

As described in Chapter 1, "[d:]" indicates the disk drive selection. However, the word "PATH" indicates something new. After the drive designation letter, you can enter the name of an existing directory and then the directory name you want to assign. It is within the specified existing directory that the new directory will be located. If you want the new directory to be created in the drive and directory you are currently working in, you don't have to enter either the drive designation letter or the path; you only enter the name for the new directory.

You can use "MD" as an abbreviated form of the MKDIR command.

For example, let's assume you have already created the three directories we used in the previous example. The resulting directory structure can be illustrated as:

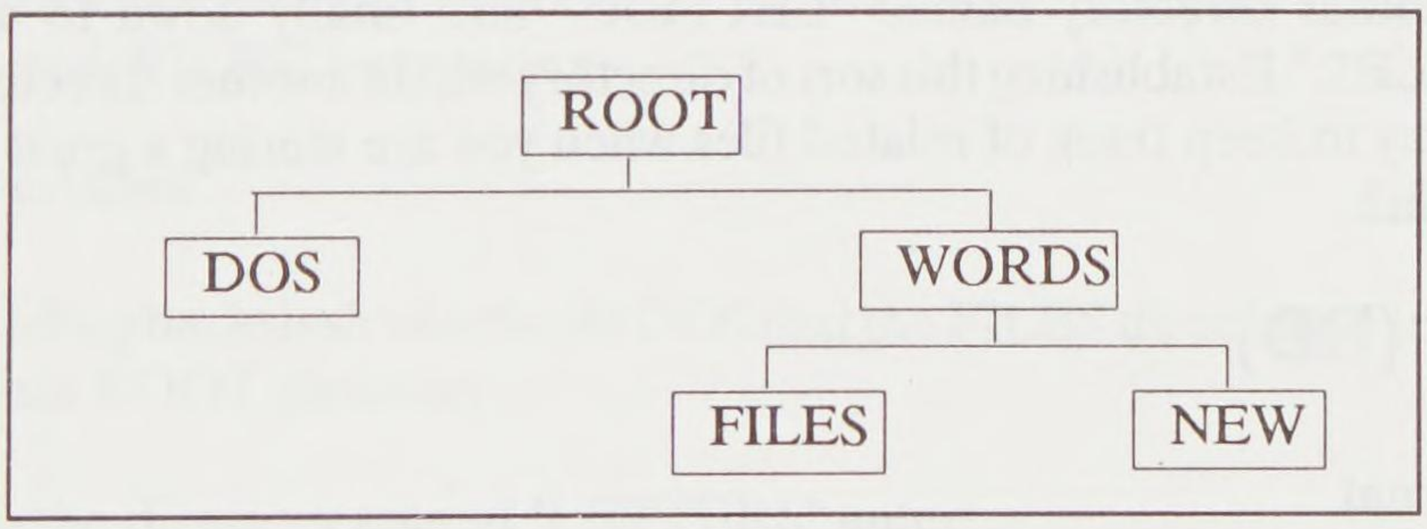


By using the MKDIR command, you can add a new directory within the root directory or to any of the existing directories. Here is how to add a new directory titled "FILES" to the "WORDS" directory.

Enter

mkdir c:\words\files

This form of the MKDIR command tells the system to create a new directory "FILES" on drive C, within the "WORDS" directory. Notice especially the backslash (\) after the C: and after the label "WORDS." Whenever you designate pathnames, DOS expects you to enter these backslashes between the names of each directory. The structure of the new directory would then look like the following:



As you can see, the "FILES" directory joins the "NEW" directory which already existed within the "WORDS" directory. If you asked DOS to display the contents of this directory, you would enter the command and the path designation as

dir c:\words

In response, DOS will display

Volume in drive C is ___

.	<DIR>	2-16-87	8:36p
..	<DIR>	2-16-87	8:36p
NEW	<DIR>	2-16-87	9:17p
FILES	<DIR>	2-16-87	9:38p

DOS Directory Indicators

Notice that even if no files exist within the specified directory, DOS will display a listing of any directories located within it. Notice also that there will always be two directory entries displayed, even if you have not stored files or other directories within it. The first directory entry is indicated by a single period (.) and the second by two periods (..). These are DOS shorthand entries: the single period stands for the current directory and the double period stands for the "parent" directory (the directory within which the current directory is located). The dates and times shown for these entries indicate when each was created. You can use these shorthand indicators in place of the name of either of these directories. For example, if you enter

cd ..

you are telling DOS to change to the parent directory.

When entering path designations, the path is indicated by the sequence of directory names alternating with backslashes. DOS reads the pathname from left to right. For example, a pathname "C:\OLDFILE\LETTERS\SALES" would indicate a path through the root directory, through a directory named "OLDFILES," on through another directory named "LETTERS" and finally down to a directory named "SALES." Establishing this sort of directory within another directory scheme is a good way to keep track of related files when you are storing a great many files on a fixed disk.

RMDIR (RD)

Type: Internal

Command Syntax:

RMDIR (RD) [d:]path

Use the RMDIR command to remove a directory. You can delete a directory only if all files (and directories) have already been removed from within it. If, for example, you wanted to remove the directory named "LETTERS" that is within the directory named "FILES," enter

rmdir c:\files\letters

This deletes the LETTERS directory, but only if it is empty. If DOS detects files (or another directory) still stored in the directory you are trying to remove, the screen displays this warning message (the specified directory will not be removed):

Invalid path, not directory,
or directory not empty

If this message is displayed, remove any files or existing directories before trying the RMDIR command again. You can use "RD" as an abbreviated form of this command.

CHDIR (CD)

Command Syntax:

CHDIR (CD) [d:]path

Use the CHDIR command to change from one directory to another.

You can use "CD" as an abbreviated form of this command.

This command uses the same type of path designations as indicated in the previous examples. For example, while working in any directory, you could change to the directory named "FILES" by entering

chdir c:\files

Notice the leading backslash which tells DOS that the FILES directory is to be found within the main ROOT directory.

To change to the directory named "LETTERS" enter:

cd c:\files\letters

Notice the second backslash which tells DOS that the LETTERS directory is to be found within the FILES directory. If you want to change from one directory to a directory within it, leave out the backslash. (The use of the first backslash indicates

a path through the root directory.) Using the same example, if you are working in the "FILES" directory and you want to change to the "LETTERS" directory, you can enter either

cd c:\files\letters

or

cd letters

The first version indicates a path that tells DOS to search (unncessarily) through the root directory. The second version does not use the leading backslash (to indicate the root directory) and therefore indicates a search path WITHIN the current directory. In this case, either version has the same effect.

PATH

Type: Internal

Command Syntax:

PATH [d:]path[;]path[d:][...]

or

PATH ;

The PATH command tells DOS where to look to find programs. You can't use this command to tell DOS where to look for text files. To specify the search pattern for text files, use the APPEND command (also described in this chapter). After you have used PATH to tell DOS where to find executable programs, you can enter the name of a program to be run even if you don't provide path information as part of a command.

Unless you use the PATH command, DOS expects to find the programs it is asked to execute in the current directory of the current drive. But, after you use PATH to specify a path designation, DOS will first search the current directory and then it will search the path you specified in the PATH command.

To understand how the PATH command is used, you need to know that if you enter a name that DOS does not recognize as one of its own internal commands, it begins to search for a program by that name; if a program by that name is found, the program is loaded and run. You use the PATH command to tell DOS where to look for such programs. If, after using PATH, you enter a name DOS does not recognize as an internal command, it refers back to the PATH information you specified and uses it to guide the search.

The PATH command guides DOS as it searches for programs that have either a COM, an EXE, or a BAT filename extension. DOS can access any program of this type in any directory on any disk, once you have specified a path with the PATH command.

For example, if you enter

```
path  \words\prog2
```

you are telling DOS, for all future searches, to automatically search the path \WORDS\PROG2 for program names you enter. This means that if you have a spell-checking program stored in the PROG2 subdirectory, you can use it from the root directory (or from any other directory) just by entering the program's name.

It's a good idea to use the PATH command to tell DOS where to find the DOS external program files. If you keep them in the DOS directory (as in some of the above examples), you do it by entering the following:

```
path  \dos
```

After you have entered this PATH command, you can use the DOS external files from any directory. However, be aware that they are executed from the DOS directory, not from the currently active directory. Therefore, for some commands, you have to be sure to enter the correct path designation for the command to act on.

You have the option of entering more than one PATH designation. You tell DOS to search more than one path by specifying multiple pathnames, each separated by a semicolon. For example:

```
path  \dos;\words\prog2
```

This form of the command tells DOS to search both the \DOS subdirectory and the \WORDS\PROG2 subdirectory.

NOTE:

It is not a good idea to specify too many paths; it will slow down your computer's response. Remember, every time you enter a PATH command, you are giving DOS another path to search: and if you accidentally enter characters that have no meaning, DOS begins to search for a program by that name. If you have specified many PATH commands, DOS continues to search every specified path until it has failed to find a match. This can take quite a while. It's better to use the PATH command only for a few important, frequently used paths (such as the path to the DOS external commands).

If you enter PATH without a path designation, DOS shows you the current path specification.

APPEND

Type: External

Command Syntax:

APPEND [/X][/E]

APPEND [d:]path[;][d:]path[...]

APPEND ;

The APPEND command, new to DOS with version 3.2, gives you a way to set the search path for data files. Use the command to tell DOS where to search for data files if a specified file is not found in the current directory. This command is similar to the PATH command that tells DOS where to search for program files (files with a .COM, .EXE, or .BAT filename extension). The APPEND command guides the search pattern for filenames, regardless of the filename extension.

Using the command followed only by a space and a semicolon (APPEND ;) deletes previously appended search paths. DOS will then search for data files only on the current drive and directory.

Use the APPEND command without options to display the current search path.

Options

/X Use the command with the /X option to expand the search pattern for data files to other paths (see example below). DOS will first search the current directory and then will search the path sequence entered with the next use of the APPEND command.

/E Use the command with the /E option to store the appended directories in the DOS environment (refer to the SET command for more information about the DOS environment). APPEND statements placed in the environment can be viewed using the SET command.

> **NOTE:**
> You can use the /X or /E parameters only the first time you use this command.

Examples

The /X switch tells DOS to first search the current drive and directory for a file you have specified, then to search paths designated using the APPEND command. Set the /X switch by entering the command in this form:

```
append /x
```

Then, if you want DOS to automatically search for files stored on drive C in either the "MEMOS" subdirectory or the "LETTERS" subdirectory that are, in turn, within another directory named SALES, enter

```
append c:\sales\memos;c:\sales\letters
```

DOS will first search for your data files in the currently active drive and directory, and then it will search the path \SALES\MEMOS, and finally the path \SALES\LETTERS. If you want DOS to search additional directories, enter this form of the command again followed by a semicolon and then enter the added path information.

> **NOTE:**
> If you plan to use the ASSIGN command to redirect drive requests, use the APPEND command first (refer to the ASSIGN command in Chapter 6 for more information).

TREE

Type: External

Command Syntax:

TREE [d:][/F]

You can use the TREE command to display the current structure of your subdirectories. If the TREE command is in the current directory (or if you have entered a PATH command to indicate to DOS the location of the command), enter

tree

DOS will report something like the following:

```
DIRECTORY PATH DISPLAY FOR VOLUME DOS1

Path is: C:
Subdirectories:

DOS
FILES
SALES

Path is: C:\DOS
No subdirectories found

Path is: C:\FILES
Subdirectories:
LETTERS

Path is: C:\SALES
No subdirectories found
```

As indicated by this example, DOS first displays all of the directories within the root directory. Then DOS lists each of the directories (if any) that are found within the directories in the first list.

Option

/F Tells DOS to also display all the files in each of the directories.

Setting Up a Fixed Disk

Before a fixed disk can be used, it must be prepared to accept data and store it according to the specific storage methods used by your computer. Computer manufacturers provide differing methods to help you carry out this set up.

While some manufacturers have gone to the trouble of preparing the fixed disk in advance, many allow you do to the setup, so you can configure the fixed disk any way you want. Often they provide a special program to help you carry out the setup tasks. The program may be named FDISK, DISKPREP, FXPREP or another similar name. These programs will usually carry out the following steps:

1. Low Level Format

This is a process that divides the tracks of the disk into the individual sectors that will eventually hold data.

2. Partitioning the Disk

This step provides a way to divide the disk in a way that allows DOS to treat a single disk as two or more physical disk drives. You can even partition the disk and use another operating system besides DOS on one part of it (for example, you could reserve a separately partitioned area for the XENIX operating system). After the disk is partitioned, you can specify that DOS address one part of the disk as Drive C, the next as Drive D, and so forth.

Most programs also allow you to specify which disk partition, if any, you want to bring into action as soon as the computer is started up (the "active" or "boot" partition).

3. High Level Format

As the final step before you can start using the fixed disk, you will also have to use the FORMAT command just as you do when preparing a diskette (some fixed disk prep programs will include this formatting process). If you plan to start your computer from the fixed disk (instead of using the system diskette), be sure to format the fixed disk using the FORMAT command with the /S option.

Refer to Chapter 6, "DOS Command Reference" for more information about using individual DOS commands with fixed disks.

Chapter 4
Using Batch Files

Batch files represent a capability of DOS that is quite different from the DOS operating system programs. A batch file is not a program; it is a set of DOS functions and commands that let you create your own programs.

Using batch files, you can tell DOS to carry out a sequence of DOS commands. In this way, you can automate the DOS commands that you find yourself using over and over.

This chapter describes how to create and use batch files. It includes many examples that you can type in and use.

What are Batch Files?

Batch files are text files; you create them just as you create any other text file. You can create batch files using your word processing program, or using the DOS COPY CON option.

Each line of a batch file is acted on by DOS in sequence: each line must be one of the following:

- A DOS batch command name
- A DOS internal or external command name
- The name of a program executable by your computer
- Another batch file name

Batch files are often referred to as programs since they combine DOS commands and application programs to carry out useful tasks.

NOTE:
Since a batch file executes one line at a time, it can be interrupted in the same way DOS commands can be interrupted: by pressing and holding the **Ctrl** key and then pressing the letter **C** key or the **Break** key. After the command is interrupted, the screen will display:

Terminate batch job (Y/N)?

At this point, you can stop the batch file by pressing **Y** (for YES) or continue by pressing **N** (for NO).

Creating a Batch File

Let's look at a simple example of a batch file. You might want to create a batch file to do two tasks in sequence: let's say that first, you want to ask DOS to display a directory of the disk that you are currently working with, and second, you want to see a display of the TREE structure of your existing directories on the disk. You can carry out these two tasks with a batch file that consists of only two commands, the DIR command and the TREE command.

You can create the batch file using your word processing program (if it can store the text as "non-document" or ASCII files). Another method is to use the COPY command, as in the example below.

To create a batch program titled "LOOK.BAT" using the DOS COPY command, type in

copy con look.bat

and press the **Enter** key.

This version of the COPY command tells DOS to copy a file named LOOK.BAT (all batch files must have the .BAT extension). But instead of copying an existing file from one place to another, this command tells DOS to copy the file from CONSOLE (the keyboard) to the current directory. In other words, you are going to create the file by typing in the commands from the keyboard.

No message will be displayed when you enter the COPY CON command. The cursor will drop down to the next line; it means that DOS is waiting for you to enter data from the keyboard.

Type in the first command:

dir

then press the **Enter** key. The cursor will drop down to the next line.

Type in the second command:

tree

and press the **Enter** key again.

Finally, to tell DOS you have finished entering data, press and hold the **Ctrl** key and then press the letter **Z** key (you can also do it by pressing the **F6** key). Then press the **Enter** key. This places an end-of-file character in the file and stores it on disk.

If you now display a directory of the current directory, you will see that you have a new file stored on the disk titled, "LOOK.BAT."

If you use the TYPE command to view the contents of the file, DOS will display

```
dir
tree
```

Your two-command batch file is ready to activate. You activate a batch file by typing in its filename (you don't have to type in the .BAT extension). In this case you would type in "LOOK" and press the **Enter** key. When you do this, the two commands DIR and TREE will be carried out in sequence: DOS will display the files in the directory you are currently using; this display will be followed immediately by a display of the tree structure of the disk.

The AUTOEXEC Batch File

One common use of a batch file is to automate what you want to take place every time you start up your computer. It's done with a special type of batch file called the AUTOEXEC batch file.

The AUTOEXEC batch file is just like any other DOS batch file with one important difference: every time DOS is loaded, it looks for the AUTOEXEC batch file in the root directory of the start-up disk. The batch file is then automatically acted on, and each command listed in the file is carried out.

If you want to carry out the same series of tasks every time you start up your computer, you can create a batch file that contains a command for each task, one command per line.

For example, the following three commands could be placed in a text file, either using your word processor or using the COPY command as described above:

```
path  c:\dos
dir
tree
```

After this file is created and stored under the named AUTOEXEC.BAT, every time DOS is loaded, these three DOS commands will be carried out in sequence. As described in Chapter 2, the PATH command will set the search path when DOS is looking for the programs you specify. The DIR and TREE commands will be activated next (they can be in the start-up directory or in the DOS directory (that was specified in the PATH command).

It is important that the file be named "AUTOEXEC.BAT," and it must be spelled correctly if DOS is to find it at startup. It must be stored on the root directory of your startup disk.

If DOS finds an AUTOEXEC batch file on the root directory of the startup disk, it will not carry out the normal operations of asking you to enter the TIME and DATE. Therefore, if you want to enter the time and date every time you start up your computer you need to enter the time and date commands as part of the AUTOEXEC batch file (as indicated in the following example:

```
time
date
path  c:\dos
dir
tree
```

This batch file will first pause to let you enter the time and date before carrying out the other three DOS commands.

You can also enter the command names of other programs in a batch file. For example, you can automatically start word processing by adding the command that starts your word processing program on one line of your AUTOEXEC.BAT file.

You can create a special batch file to load a series of programs. If, for example, you wanted to ask DOS for a display of files on the disk that end in .TXT, then load a memory-resident spell checking program called SPELLER, and finally load a word

processing program called EDITOR, you could create a batch file to do it. Let's create the file and title it "LOAD.BAT."

The contents of the file would look like this:

```
dir  *.txt
speller
editor
```

After you create the file and store it on disk, all you have to do is enter the word "LOAD" and press the **Enter** key. The three tasks will be carried out in sequence.

If you remove a disk while a batch file is running, DOS will ask you to reinsert it when the batch file finishes acting on a command and is ready to act on another. DOS reads from the batch file for each command. If it does not find the file in the drive, it will display the following message:

Insert disk with batch file and press any key when ready

Put the disk back into the drive and press a key.

> **NOTE:**
> If you do not want to take time to put the batch file disk back into the drive, remember, you can interrupt any batch file at any time by pressing the **Ctrl** + **C** key combination.

Passing Parameters to Batch Files

One other very useful feature of batch files is their ability to use a parameter that you "pass" on the command line (at the DOS prompt) at the time you invoke the file. The parameter you enter on the command line (after the name of the batch file) can then be used in the batch file.

As we have stated, you start a batch file by entering its name on the command line (with or without the .BAT filename extension). If you also enter other characters on the command line after the batch file name (always leave a space after the batch file name), DOS stores them in memory for later use in the batch file; in this way, they are "passed" through to the batch file and are usable as parameters.

You can enter just about any string of characters on the command line after the batch file name (but don't use the file redirection characters ">" or "<"). With most versions of DOS, you can use a string of up to 98 characters in length (with no spaces) and it will still be passed to the batch file as a single parameter.

You can enter as many of these parameters on the command line as you want (each separated by one or more spaces), but you will have to use the SHIFT command in your batch file to be able to access more than ten of them (the SHIFT command is described later in this chapter).

By passing parameters from the command line you can create batch files that will operate conditionally: your batch file can test to determine what was entered on the command line. Within the batch file, you specify the passed parameter by using a percent (%) sign and a number. The first string of characters you enter after the name of the batch file will be stored in the computer's memory and can be referred to within the batch file as %1. For example, let's assume you are starting a batch file named "LOAD.BAT." If you enter

```
load start
```

and press the **Enter** key, the LOAD.BAT batch file will be started and the word "start" will be temporarily placed in memory. If, when you created the batch file, you had used the symbol "%1" on any line of the batch file, the %1 will be replaced by the word "start" while the batch file is in operation. Typically, your batch file might contain a line to determine what characters (if any) were entered on the command line after the name of the batch file. For example, let's say the following line appears in the LOAD.BAT batch file:

```
If %1==start goto STARTLINE
```

The %1 symbol will be replaced by the word "start" when the batch file is in operation, resulting in the line

```
If start==start goto STARTLINE
```

The line compares the two words and the GOTO batch command will be carried out (the batch commands are described later in this chapter).

If, on the other hand, no additional characters were entered when the batch file was started, the line would appear in the batch file as

```
If == start goto STARTLINE
```

There is no equality in the line and therefore the GOTO command will not be acted on.

The name of the batch file itself is stored as %0. Therefore, if you start the batch file by entering

```
load
```

and the batch file includes the line

```
ECHO The name of this batch file is %0
```

during the execution of the batch file, the line would change to read

```
The name of this batch file is load
```

The next nine strings DOS finds after the batch file name are stored as %1 through %9.

As a more useful example, let's modify the example LOAD batch file that was described in the last section so that it uses a passed parameter. As you will recall, the file contained three DOS commands that carried out three tasks in sequence, the last of which was to load the word processing program titled "EDITOR." But there is a way to add the powerful batch file feature of acting on a passed parameter: you can change the LOAD batch program to make it capable of executing any file you name as a parameter when you start the batch file.

To change the LOAD batch program to act on a passed parameter, you would first recreate the file by entering the same two DOS commands as you did when creating the LOAD.BAT file:

```
dir *.txt
speller
```

But instead of also adding the last line "EDITOR" as we did in the previous example, this time enter

```
%1
```

This will create a new version of the LOAD.BAT that looks like this:

```
dir *.txt
speller
%1
```

When this batch file is run, DOS will first display all of the files that end in .TXT and then, as before, load the program titled "SPELLER" into memory. But now, the third task is the %1 symbol for the passed parameter. This tells DOS to act on whatever name was entered on the command line after the batch file name. If you start the batch file by entering

```
load  editor
```

when the program runs it will be acted on as

```
dir  *.txt
speller
editor
```

and the program named "EDITOR" will be loaded and executed.

If you enter the name of any other program on the command line (after the name of the batch file), it will be that program that is acted upon as the first parameter (%1). If the third line of the batch file is to work properly, the parameter you enter on the command line must be the name of a DOS command or an executable program. It must also be available in the current directory (or else via the path indicated by a PATH command). As you can see, using the passed parameter gives your batch files much more power and flexibility.

You can use more than one passed parameter at a time. For example, we can change the LOAD batch file to make it even more powerful by setting it up to receive several passed parameters. You might create this new "LOAD.BAT" batch file:

```
dir  *.txt
%1
%2
%3
```

With this form of the LOAD program, you can tell DOS to load and run one, two, or three programs that are specified at the time you call the LOAD program. If you start the LOAD program by entering

```
load  speller  editor
```

the batch file tells DOS to first display the directory of files with the .TXT extension, then load the program named "SPELLER" (recognized by the program as the first passed parameter, %1), and then load the program named "EDITOR" (recognized by the program as the second passed parameter, %2). Since no third string was entered, the %3 is ignored. Likewise, if you do not enter anything after the

name "LOAD," the program will take no action when it finds the numbers followed by the percent sign. Whenever a batch program encounters a percent symbol followed by a number, it checks to see if a parameter is in memory (if characters were entered on the command line). If no parameter is currently in memory, then the percent symbol and number are ignored as the batch file is executed. In this case, if no characters were entered on the command line, the first step of the program will be carried out and the program will end.

Given this example, you can probably think of other DOS tasks that can be carried out more quickly and simply by using batch files with passed parameters.

Batch Command Summary

DOS provides some special commands that can be used in batch files. They are summarized in the following sections. Complete descriptions of each of these commands can also be found in Chapter 6, "DOS Command Reference."

CALL

Type: Internal

Syntax:

CALL [d:][path]batchfilename [options]

The CALL command, new to DOS in version 3.3, calls another batch file and then returns to the current batch file to continue.

You can use CALL within a batch file to specify the name of another batch file (any file with the .BAT filename extension). The original batch file calls into action the second batch file; when the called batch file ends, control is transferred back to the original batch file.

The CALL command provides a function – transfer of control back to a calling batch file – that was not available prior to MS-DOS version 3.3.

Example

To run the files STARTER.BAT, TESTER.BAT, and FINISH.BAT in sequence, enter the following three lines in your batch file:

```
call starter
call tester
call finish
```

When you run this batch program (and if the three test files contain only an "ECHO OFF" command and an identifier line), DOS will display

```
call starter

echo off
This is starter.bat
call tester

echo off
This is tester.bat
call finish

echo off
This is finish.bat
```

ECHO

Type: Internal

Syntax:

ECHO ON ¦ OFF

ECHO (message)

The ECHO command turns on or off the display of commands in a batch file. Normally, all commands in a batch file are displayed as the batch file runs. ECHO OFF turns off this feature. ECHO ON turns the display feature back on. By using this command in your batch files you can turn the display feature on and off one or more times as the file runs.

If you enter this command with no options, DOS will display the current ECHO setting.

With some versions of DOS, you can force DOS to display a blank line as the batch file carries out the sequence of commands by entering "ECHO" alone on a line. With other versions, you must enter some recognizable character after "ECHO" for the command to function properly.

Example

If you want to display the message "This is file one" during the execution of a batch file, but you do not want to display any of the commands being acted upon, enter

```
echo  off
echo  This is file one
```

When you run this batch program, DOS will display

```
echo  off
This is file one
```

As indicated above, the words "This is file one" will be displayed when processing acts on that line in the batch file. The words "echo off" on the first line will be displayed, but the word "echo" on the second line will not.

Notice that you do not have to surround the message with quote marks as is required in BASIC and other programming languages.

FOR

Type: Internal

Syntax:

FOR %%(variable) IN set DO (command)

The FOR command performs repeated execution of commands (for both batch processing and interactive processing). It provides a way to repeat DOS commands in a predetermined way (usually to tell DOS to carry out a command on each of the files named in the set specifications).

%%variable can be any character. *set* can be one or more file specifications.

The *%%variable* is set sequentially to each of the set entries, then the DOS command that was specified is carried out. Typically *set* will include wildcard characters (? and *) to indicate a group of filenames. Then, *%%variable* is set to each matching filename. You can also use path designations for *set*.

Examples

To TYPE (display) the contents of every file in the current directory that has the filename extension ".TXT," enter

```
for %%X in (*.txt) do type %%X
```

If DOS finds files on the disk that have the TXT extension, the command sets %%X to each of the .TXT files then displays the contents of each file.

Let's assume you have stored the following three files on the disk that have the TXT extension:

MON.TXT
TUES.TXT
WED.TXT

The example line in the batch file will carry out the following steps:

First, the command sets %%X equal to each of the files specified inside the parenthesis. In this case it specifies

```
%%X in (*.TXT)
```

Since the files specified inside the parenthesis are all the files that have the TXT extension, the command would first set %%X equal to the first listed file that ends in TXT (in this case, the file "MON.TXT").

Second, the next part of the command, stated as

```
do type %%X
```

tells DOS to TYPE out the contents of the %%X file. Since the %%X has now been set to "MON.TXT," the line will be acted on as

```
type mon.txt
```

After the TYPE process has been carried out, the entire procedure will be repeated again until all the files ending in ".TXT" have been acted on. DOS will TYPE the TUES.TXT file and then the WED.TXT file. When the command finds no more files, it terminates.

When the line is executed, if the .TXT files contains only the "ECHO OFF" command and one identifying line, DOS will display

```
for %%X in  (*.txt) do type %%X

type MON.TXT
echo off
echo This is mon.txt

type TUES.TXT
echo off
This is tues.txt
type WED.TXT

echo off
This is wed.txt
```

GOTO

Type: Internal

Syntax:

GOTO (label)

The GOTO command causes an unconditional branch to the specified label and processing begins again with the line found immediately after the label.

A label is a string of eight or fewer characters preceded by a colon. It must be the first characters on the line, but it can be indented. The label can include spaces, but it cannot include restricted characters (refer to Chapter 1, "Introduction" for a discussion of restricted characters).

DOS will read the first eight characters that you specify after the GOTO command as the label. You can use a label of more than eight characters, but DOS will not recognize any characters after the first eight.

For example, DOS will act on this line

```
goto  startofthisfile
```

by looking for the label, ":STARTOFT" on any other line of the batch file.

A label can be of either upper or lower case characters; DOS does not distinguish between them.

Example

To create a batch file that will function as an infinite loop, enter

```
echo  off
:START
echo  This is an infinite loop
goto  START
```

The first line of the batch program turns off the echo display feature. It takes no action on the next line (:START) and the third line displays the message, "This is an infinite loop."

When you enter the name of this batch file, DOS will display

```
echo off
This is an infinite loop
This is an infinite loop
This is an infinite loop
This is an infinite loop
This is an infinite loop
This is an infinite loop
This is an infinite loop
This is an infinite loop
This is an infinite loop
This is an infinite loop
This is an infinite loop
This is an infinite loop
This is an infinite loop
This is an infinite loop
This is an infinite loop
This is an infinite loop
This is an infinite loop
This is an infinite loop
```

The batch program acts on the GOTO command by looking for a line beginning with the label "START." If the label is not found (or if it includes illegal characters) DOS will display

Label not found

If a the label is found, DOS transfers control to the line following the label. In this case, the GOTO command transfers control back to the beginning of the file, to the line following the ":START" label.

The file will continue this cycle until terminated (you can press the **Ctrl** + **C** key combination to stop a batch file at any time).

This is not an especially useful application of the GOTO command, but is intended to illustrate its function. Refer to the next section describing the IF batch file command for a useful application of the GOTO command.

IF

Type: Internal

Syntax:

IF [NOT] EXIST filename (command)

IF [NOT] (string1)==(string2) (command)

IF [NOT] ERRORLEVEL (number) (command)

When used in combination with other DOS commands, the IF command provides a way to conditionally carry out DOS commands. This command makes DOS batch files much more useful because it provides for options within the batch file.

Using the IF command, a condition can be tested to determine if it is true (or NOT true) and a DOS command can be issued depending on the outcome of the test.

As indicated in the command syntax, the IF command is somewhat more complicated than most DOS commands. The conditional test can vary considerably in form and can be used for a number of different types of tests. The conditional tests can be grouped into the three types of conditions summarized in the sections that follow.

Condition 1: IF [NOT] EXIST (file specification) command

This condition is determined by a test to see if a file exists (or does not exist) on disk.

When this conditional test is included (it can be on any line of your batch file), DOS checks to determine if a file exists (you must specify the path so DOS can find the file). If the condition is true (as stated), the specified command is executed. If the condition is found to be false, the command is not acted on and DOS proceeds to the next line in the batch file.

When the IF command is used together with the GOTO command, it provides a way to transfer control to a different line in the batch file dependent on the outcome of the test.

Examples

For example, to display a warning message only if the file "DATA.1" does not exist in the current directory, enter

```
echo off
if not exist data.1  goto   MESSAGE
echo  The file DATA.1 exists on the current directory
goto  END
MESSAGE
echo  The file DATA.1 was not found.
echo  The file must be in the current directory
:END
```

When you enter the name of this batch file, there are two possible outcomes. If the file DOES NOT EXIST in the current directory, DOS will display:

```
echo off
The file DATA.1 was not found.
The file must be on the current directory
```

If the file DOES EXIST in the current directory, DOS will display:

```
echo off
The file DATA.1 exists on the current directory.
```

In this example, the IF command is combined with the GOTO command to determine the optional branching. The command tells DOS to check the current disk to determine if the file DATA.1 exists. Since we are using the NOT EXIST form of the command, the condition is true if it DOES NOT find the file; since the condi-

tion is true, the specified command (in this case, the GOTO command) is acted on. The "GOTO MESSAGE" command passes processing to the ":MESSAGE" label and the messages on the following lines are displayed.

Condition 2: IF [NOT] string1==string2 command

This operation checks to determine if the first string you enter is equal (or is NOT equal) to the second string you enter. If the condition is true (as stated), the specified command will be carried out.

DOS judges this condition based on whether or not the two strings are EXACTLY identical. DOS WILL detect a difference between uppercase and lowercase letters.

Example

We can create a very useful version of the IF command that checks to determine whether of not a parameter was passed to the batch file from the command line. The batch file below acts conditionally depending on a test of whether or not any characters were entered on the command line after the batch file name.

```
echo off
if not x==%1x goto  PRINT
echo  You must enter a parameter immediately after batch file name
goto  END
:PRINT
echo  The passed parameter is [ %1 ]
:END
```

The second line of this useful batch file checks to determine whether of not a parameter has been passed at the command line (for more information on passing parameters to a batch file, refer to the section by that title earlier in this chapter). It uses the IF command to determine if string one (the character 'x') is equal to string two (the character 'x' plus the %1 symbol that represents the first passed parameter). If a parameter WAS passed to the batch file, the two strings will not match and the condition (stated as NOT equal) is true; the GOTO command will be acted on, transferring processing to the label ":PRINT."

If the condition is found to be false (the two strings ARE equal), it means a parameter WAS NOT entered on the command line. In that case, the GOTO command on the same line as the IF command will not be acted on and processing will be transferred to the next line of the batch file. The next line includes the message "You must enter a parameter immediately after batch file name." The following line transfers the processing to the ":END" label and the program terminates.

If characters WERE entered after the batch filename, DOS will display

```
echo  off
The passed parameter is [ test ]
```

If characters WERE NOT entered after the batch filename, DOS will display

```
echo  off
You must enter a parameter immediately after batch file name
```

Condition 3: IF [NOT] ERRORLEVEL (number) command

This operation checks to determine if an ERRORLEVEL number is set (or is NOT set) at or greater than the specified value. The ERRORLEVEL number can be set by application programs and the value is retained in the computer's memory. This command is used to check that value.

> **NOTE:**
> DOS judges this condition to be true if the ERRORLEVEL code is equal to OR GREATER than the number you specify.

Example

A common use of the IF command is to display a message to indicate that an ERRORLEVEL has been set. To display a message whenever any ERRORLEVEL number has been set (any value above zero), enter:

```
echo off
if  ERRORLEVEL  1  echo  The errorlevel is now greater than zero
```

When this line is acted on (and if the ERRORLEVEL is found to be set to a value), the ECHO command will be acted on and DOS will display

```
echo  off
The errorlevel is now greater than zero
```

If DOS does not find a value set for ERRORLEVEL, the program terminates without displaying the message.

SHIFT

Type: Internal

Syntax:

SHIFT

The SHIFT command provides a way to increase the number of replaceable parameters to more than the standard ten. (For more information, refer also to the section titled "Passing Parameters to Batch Files" earlier in this chapter.)

As stated earlier in this chapter, DOS provides a way to pass strings of characters as parameters to the batch file. For this use, it reserves the symbols %0 through %9. Therefore, this set of ten symbols limits you to ten parameters that can be passed. But even with this limitation of ten symbols, the SHIFT command can be used to shift the assigned parameters to make room for the use of additional parameters.

For example, if you started a batch file named "GO.BAT" by entering the following on the command line:

go par1 par2 par3 par4 par5 par6 par7 par8 par9 par10

The parameters available for use in the batch file "GO" will be set as follows:

%0 = go (name of batch file)
%1 = par1
%2 = par2
%3 = par3
%4 = par4
%5 = par5
%6 = par6
%7 = par7
%8 = par8
%9 = par9

To use the additional (eleventh) parameter you entered on the command line (par10), enter

shift

The parameters will be reset as

%0 = par1
%1 = par2
%2 = par3
%3 = par4
%4 = par5
%5 = par6
%6 = par7
%7 = par8
%8 = par9
%9 = par10

The set parameters are shifted one higher, giving you access to more than ten parameters. Each time you enter the SHIFT command, the parameters used by the batch file is shifted up one position.

> **NOTE:**
> Notice that each time you use the SHIFT command in a batch file, the lowest numbered parameter will be lost. It can no longer be referenced in the batch file.

PAUSE

Type: Internal

Syntax:

PAUSE [comment]

The PAUSE command suspends execution of a batch file until a key is pressed. Execution of the batch file is temporarily suspended at the point at which it appears in the file. After suspension of the batch file, DOS will display the message

Strike a key when ready ...

At this point you can press any key to restart the program (or use the **Ctrl** + **C** key combination to stop the program).

On some versions of DOS a comment may also be entered on the same line as PAUSE. The comment prompt is displayed before the "Strike a key when ready" message.

This command is useful if you want to pause the execution of a batch file and place control of its restart in the hands of the batch file user. It is most commonly used to give the user time to read the information on the screen or to temporarily suspend processing until a new diskette is inserted.

To suspend execution of a batch file and display the message "Insert the target diskette in drive A," enter

```
echo off
echo  Insert the target diskette in drive A
pause
```

When the batch file processes these two lines, DOS will display

```
Insert the target diskette in drive A
Strike a key when ready ...
```

REM (Remark)

Type: Internal

Syntax:

```
REM  [comment]
```

The REM command provides a way to insert remarks (that will not be acted on) into a batch file. During execution of a batch file, DOS will not act on text or commands which are entered on the same line as the REM command in a batch file. DOS will, however, display the remarks unless you use the ECHO OFF command to suppress this display.

You cannot use separators in the comment except the space, tab, and comma. To keep DOS from interpreting commands in a comment line, enclose the command in quotes.

To add a comment to a batch file, enter

```
REM  This is a comment line
```

To keep the REM comment line from being displayed as the batch file is run, enter

```
echo  off
rem  This comment line will not be displayed
```

Advanced Batch Techniques

Once you have learned to create the useful types of batch files described in this chapter, you may find yourself creating batch files "on the fly." That is, if you are like many advanced computer users, you will use the DOS batch file capabilities to carry out many of your most commonly used file and disk management tasks; you can quickly create short batch programs to do some required, but repetitive, task.

For example, let's say you have frequently been loading the spell checker "SPELL" that requires the special dictionary file "WORDS2" and then loading the word processor "EDIT" and the file "TAXES.87." You can automate the entire procedure with this simple batch file,

```
cd\editor
spell words2
edit taxes.87
```

The batch file will change to the appropriate directory and then load the files. If you name the batch file something simple to enter like "TAX," you can enter the word "TAX" immediately after you start your computer and be ready to go to work.

You can create one of these task-oriented batch files for each of your most common DOS tasks. Keep these batch files (and the external DOS files) in a special directory that is named with the PATH command in the AUTOEXEC.BAT file. That way, DOS can find the batch file and act on it from any directory on any disk drive.

The section that follows provides a variety of suggestions for creating batch files that can be used to carry out typical DOS tasks.

Useful Batch Files

A typical use of batch files is to help you quickly carry out the DOS tasks that are used over and over. Among the simplest are a set of one "letter" batch files that are used for disk handling shortcuts. Keep these batch files in your batch file directory that is named in the PATH command in the AUTOEXEC.BAT file.

Below, we have reproduced the exact text for some of these "quicky" batch files. To make use of them, use your word processor to enter them exactly as they are printed here. Make sure they are stored on disk as "standard" ASCII files (without special file formatting characteristics). They can even be entered using the COPY CON option as described earlier in this chapter.

Batch Filename: C.BAT

C.BAT

```
echo off
cd\%1
%2
%3
```

Using C.BAT

You can enter the letter C followed by any path designation to quickly change to a new directory on the current drive. You can also pass one or two additional parameters at the same time. These options can be the name of a word processor or any other executable program that is accessible from the directory you are switching to. If you do not enter the options for %2 or %3, they will only pass through an "empty" carriage return.

For example, if you enter

c letters edit

the batch file will change to the directory "\LETTERS" and start the word processing program "EDIT." Notice that you do not have to enter the backslash (\) because it is included in the batch file.

Batch Filename: D.BAT

D.BAT

```
echo off
dir %1/p
or
echo off
dir %1/w
```

Using D.BAT

You can enter the letter **D** followed by any path designation to quickly display a directory listing. The first version will display the file list one screen at a time; the second version will display them in the five column format.

For example, if you enter

```
d  \letters\*.txt
```

the batch file will display all of the files in the "\LETTERS" directory that have the .TXT filename extension. Notice that this time the backslash (\) is not included in the batch file; it must be entered as part of the path designation.

Advanced Batch Files

As seen in the previous examples, batch files can be used to carry out a number of common computer tasks. But batch files can also be used for some very complicated computer tasks. In this section, we offer some suggestions for creating a variety of powerful batch programs. Since many of the batch file tasks require at least a working understanding of programming techniques, we suggest working through several of the examples presented earlier in this chapter before undertaking their creation.

Refer also to Chapter 5, "Tips for Advanced Users" for examples of additional techniques using batch files.

Batch Filename: FORMAT.BAT

FORMAT.BAT

```
echo off
if %1x==x goto MESSAGE
if %1 == A: goto START
if %1 == a: goto START
if %1 == B: goto START
if %1 == b: goto START
goto WARNING
:START
formatx  %1  %2  %3  %4  %5  %6
goto END
:WARNING
echo You entered [ %0  %1  %2  %3  %4  %5  %6 ]
echo You can only format drives A and B using this program
goto END
:MESSAGE
echo No parameter entered.  You must enter FORMAT (drive):
:END
```

Using FORMAT.BAT

Some versions of DOS allow you to use the FORMAT command to reformat the fixed disk. If your DOS version allows this, you could accidentally erase all the data on the fixed disk (for example, by entering the FORMAT command with no drive designation while you are working in drive C). This batch file protects the fixed disk from accidental format by checking the drive identification letter before acting. This example is for a system that includes two diskette drives, A and B.

Before using this batch file, rename the FORMAT.COM file to FORMATX.COM.

Two different ccomparisons are carried out. The second line of the batch file is

```
if %1x==x goto MESSAGE
```

This line uses the "IF string one = = string two" form of the IF command conditional test to determine whether or not any characters were entered on the command line after the name of the batch file. It is a check to be sure a drive identification letter was entered. If not, it transfers processing to the label "MESSAGE" which displays

```
No parameter entered.  You must enter FORMAT (drive):
```

Next, the batch file checks to be sure the identified drive is either A or B (it checks for both upper and lower case letters). If the passed parameter is not A or B, the batch file displays what the user entered on the command line and then displays the warning "You can only format drives A and B using this program." However, if A or B was entered, the renamed format file (FORMATX.COM) is called and the batch parameter is passed to it. The rest of the formatting procedure will proceed as usual and the batch file will end.

NOTE:

If you DO want to format the fixed disk, you can still use the renamed DOS format program to do it. For example, to format drive C, enter

```
formatx  c:
```

DOS will display the usual prompts and then the format process will be carried out.

Batch Filename: BACK.BAT

BACK.BAT

```
echo off
if %1x==x goto MESSAGE
goto START
:MESSAGE
echo  No parameter entered
echo  Enter a date (backup files created after this date)
echo  Use the format   mm-dd-yy
echo  Terminating program
goto END
:START
echo  Ready to back up to drive A all files on the fixed disk that
echo  were created or altered on or after the following date: [ %1 ]
echo  Press any key to continue or press Ctrl + C to cancel
pause
mcopy c:*.* /s /d:%1 a:
:END
```

Using BACK.BAT

This batch file is useful to selectively back up files from your fixed disk to the diskette in drive A.

As in the FORMAT.BAT example above, the batch file first checks to make sure a parameter was entered on the command line (the BACK.BAT file uses this parameter to selectively back up all files after the date specified).

If a parameter was not entered, the batch file displays instructions (at the label ":MESSAGE") and the GOTO END ends the program. If a parameter was entered, the program displays a message (at the label ":START") explaining the backup procedure that is about to be carried out. The PAUSE command gives the user a chance to read the message and cancel, if desired, before the backup begins.

The actual backup procedure is carried out using the MCOPY version of the XCOPY command with the /S and /D options (for more information on the XCOPY command, refer to Chapter 5, "Tips for Advanced Users" or Chapter 6, "DOS Command Reference").

The /S option tells DOS to copy all directories and subdirectories (unless they are empty). The /D option is used with a date passed as a parameter to specify files created on or after the date entered.

Therefore, if you enter

back 5-21-88

All files created after the date "5-21-88" will be backed up to the disk in drive A.

You can modify this batch file to selectively back up files using other XCOPY options.

Interactive Batch Files

Thus far, we have presented batch files that act selectively, based on a parameter that is passed to the batch file from the command line.

A factor that greatly limits the use of batch files is lack of a DOS command that can be used to take input from the user while a batch file is in operation. If such a command were available, it would allow conditional actions within the batch file based on the keys pressed by the user.

Although there is no such DOS command, we can create our own. It is done using the DEBUG program which is usually supplied with DOS.

Creating GET.COM

The GET.COM program places a number related to a key press in the area of the computer's memory that can be accessed by DOS as the ERRORLEVEL parameter (for more information about the ERRORLEVEL command, refer to the section by that name in this chapter).

To create the file GET.COM, insert the diskette (or switch to the directory) that contains the DEBUG.COM program. Enter

debug get.com

After the DEBUG program is loaded, it will display

File not found

This message indicates that we have not yet created the file named GET.COM (we will create it at the end of this process). The dash (–) is the command line prompt for DEBUG (it's like the A prompt that DOS normally uses).

Right next to the dash, enter

```
a100
```

the program will display some numbers. Without moving the cursor from its position next to the numbers, enter each of the four lines below and press the **Enter** key after each line. Each time you enter a line and press the **Enter** key the program will display a new set of numbers. You enter each of these lines right next to the number that is displayed.

```
mov ah,00
int 16
mov ah,4c
int 21
```

After you enter the last line, the program will again display a set of numbers. This time, you only have to press the **Enter** key (to tell the program you have finished entering data). The program will again display the dash. Now enter

```
rcx
```

and press the **Enter** key. The program will again display numbers and then a colon. Enter

```
8
```

When the dash appears again, enter

```
w
```

The program will display

```
Writing 0008 bytes
```

This means the program has now been created and written to disk. When the dash again appears, you can quit the DEBUG program by entering

```
q
```

If you now look at the files stored on the disk, you will find a new file, "GET.COM." You can use it in your batch files to take a one keystroke input from the user. The next section provides some examples of its use.

Creating Menus

Once you have created the GET.COM program, you can use it in your batch files to take input from the computer user. The GET program accepts any keystroke from the keyboard and stores the ASCII value in the ERRORLEVEL area of memory. By checking the current value stored in that area, we can determine what key the batch file user pressed.

The following sample program uses the GET.COM program to take a user selection from a menu. You can modify this program to suit your own purposes.

Batch Filename: MENU.BAT

MENU.BAT

```
echo off
cls
echo
echo  Menu Options
echo
echo
echo
echo 1  WORD PROCESS
echo 2  SPREADSHEET
echo 3  MAINFILE
echo 4  DATABASE
echo 5  PROGRAMMING
echo 6  MOUSE PROGRAMS
echo 7  DOS PROGRAMS
echo 8  SERVICE GUIDE
echo 9  EXIT TO DOS
echo
echo  Type in the number for your choice
echo
:GET1
get
rem Value higher than 58 is in error so get another number
if errorlevel 58 goto GET1
rem test for key value 57 (ie, 9 the EXIT option)
if errorlevel 57 goto EXIT
rem Now test for key value for 8 key (ie, the EIGHTH option)
if errorlevel 56 goto EIGHTH
rem Now test for key value for 7 key (ie, the SEVENTH option)
if errorlevel 55 goto SEVENTH
rem Now test for key value for 6 key (ie, the SIXTH option)
if errorlevel 54 goto SIXTH
rem Now test for key value for 5 key (ie, the FIFTH option)
```

```
if errorlevel 53 goto FIFTH
rem Now test for key value for 4 key (ie, the FOURTH option)
if errorlevel 52 goto FOURTH
rem Now test for key value for 3 key (ie, the THIRD option)
if errorlevel 51 goto THIRD
rem Now test for key value for 2 key (ie, the SECOND option)
if errorlevel 50 goto SECOND
rem Now test for key value for 1 key (ie, the FIRST option)
if errorlevel 49 goto FIRST
rem Check for value of 48 or greater (get another number)
if errorlevel 48 goto GET1
:FIRST
cd\wp
edit
goto EXIT
:SECOND
cd\spread
spread
goto EXIT
:THIRD
cd\mainfile
edit
goto EXIT
:FOURTH
cd\database
data1
goto EXIT
:FIFTH
cd\program
prog
goto EXIT
:SIXTH
cd\mouse
mouse
goto EXIT
:SEVENTH
cd\dos
goto EXIT
:EIGHTH
cd\service
edit svguide.doc
:EXIT
```

Using MENU.BAT

This batch file displays nine choices from a formatted menu that fills the screen until a valid choice is made. When you enter the name of the batch file, the screen is cleared and the menu appears. It should look like this:

```
Menu Options


1  WORD PROCESS
2  SPREADSHEET
3  MAINFILE
4  DATABASE
5  PROGRAMMING
6  MOUSE PROGRAMS
7  DOS PROGRAMS
8  SERVICE GUIDE
9  EXIT TO DOS

Type in the number for your choice
```

When the user starts the program, the screen is cleared and the menu is displayed. Next, the line "get" calls the GET.COM program into action. The program will not continue until the user presses one of the number keys.

The batch file first checks the ERRORLEVEL setting for a value of 58 or higher (the ASCII value of the number 9 key is 57). If the value is higher than 58, it indicates that a number key was NOT pressed (all letter keys have values higher than 58). In that case, the GOTO command transfers processing back to the label "GET1" and the GET.COM program is again called into action and waits for another key to be pressed.

After the possibility of a value 58 or higher has been checked, the next IF test looks for an ERRORLEVEL value of 57. If found, it means that the number 9 key was pressed (since all higher values have been checked) and processing is transferred to the label ":EXIT" and the program ends.

Since all values greater than 57 have been eliminated, the program can test for lower values to determine which key was pressed. You must always eliminate higher values first because the IF test checks for the specified value OR GREATER.

In each of the IF tests that follow, processing is either transferred to a corresponding label (where one or more programs are activated) or else the program continues to the next line. The last test checks for the value 48 (the ASCII value for the zero key). If this value is set (or any lower value) processing is sent back the the GET1 label to wait for another key press.

Getting Yes/NO Input

The GET.COM program can also be used in a batch file to take a **Y** or **N** input from the user. The program also uses the ERRORLEVEL setting and carries out a series of IF tests to determine if the **Y** or **N** key was pressed.

Batch Filename: LOAD-YN.BAT

LOAD-YN.BAT

```
echo off
cls
echo Load EDIT program?  (Enter Y or N)
:GET1
get
rem Test for 122 or greater (value above y)
if errorlevel 122 goto GET1
rem Test for 121 (y)
if errorlevel 121 goto YES
rem Test for 111 (value above n)
if errorlevel 111 goto GET1
rem Test for 110 (n)
if errorlevel 110 goto NO
rem Test for 90 (value above Y)
if errorlevel 90 goto GET1
rem Test for 89 (Y)
if errorlevel 89 goto YES
rem Test for 79 (value above N)
if errorlevel 79 goto GET1
rem Test for 78 (N)
if errorlevel 78 goto NO
rem For any other key goto GET1
goto GET1
:YES
```

```
echo Y key pressed - loading EDIT
edit
goto EXIT
:NO
echo N key pressed - exiting program
:EXIT
```

Using LOAD-YN.BAT

You can use LOAD-YN.BAT to carry out a YES/NO branch within a batch file. In this example, if the user presses the **Y** key, the program displays

```
echo off
Load EDIT program?  (Enter Y or N)

Y key pressed - loading EDIT
```

Like the MENU.BAT file, this program uses the GET.COM program that was described in the previous section and a series of IF tests to check the ERROR-LEVEL. As in MENU.BAT, this program first checks for a value higher than 121 (the ASCII value for the **y** key). If no higher value was set, the program checks for the 121 value. If found, it means the **y** key was pressed and the processing is transferred to the ":YES" label where a message is displayed and the specified program is loaded.

Next, the program checks for a value higher than 110 (the value for the **n** key) and then for the value of the key itself. If this value is found, the processing is transferred to the ":EXIT" label and the program ends.

This series of IF tests continues for the same two letters in uppercase.

If a value is not found for any of these keys, the program uses the GOTO command to branch back to the ":GET1" label to again execute the GET.COM program that waits for another key press.

A Powerful AUTOEXEC.BAT File

The GET.COM program that we created using the DEBUG program (described in the previous sections), provides a way to take input from the user while a batch file is running. This also gives us a way to take input during the system's startup using the AUTOEXEC.BAT file.

The following example is somewhat complicated, but if you enter similar lines in your AUTOEXEC.BAT file, you can create a program that will give you a great deal of choice about your system configuration when you start your computer.

Batch Filename: AUTOEXEC.BAT

AUTOEXEC.BAT

```
echo off
path = c:\dos;c:\
append c:\words
rem color version of prompt
prompt $e[s$e[1;67H$d$e[2;67H$t$h$h$h$e[1;31;40m$e[u$p$e
    [0;32;40m$g
echo
echo
echo  Show options menu (Y/N)?
echo
echo
:GET1
get
rem Test for 122 or greater (value above y)
if errorlevel 122 goto GET1
rem Test for 121 (y)
if errorlevel 121 goto YES
rem Test for 111 (value above n)
if errorlevel 111 goto GET1
rem Test for 110 (n)
if errorlevel 110 goto EXIT
rem Test for 90 (value above Y)
if errorlevel 90 goto GET1
rem Test for 89 (Y)
if errorlevel 89 goto YES
rem Test for 79 (value above N)
if errorlevel 79 goto GET1
rem Test for 78 (N)
if errorlevel 78 goto EXIT
rem For any other key goto GET1
goto GET1
:YES
cls
echo
echo  Menu Options
echo
echo
echo  1 WORD PROCESS
```

```
echo  2 SPREADSHEET
echo  3 MAINFILE
echo  4 DATABASE
echo  5 PROGRAMMING
echo  6 MOUSE PROGRAMS
echo  7 DOS PROGRAMS
echo  8 SERVICE GUIDE
echo  9 EXIT TO DOS
echo
echo  Type in the number for your choice.
echo
:GET2
get
rem Value higher than 58 is in error so get another number
if errorlevel 58 goto GET2
rem test for key value 57 (ie, 9 the EXIT option)
if errorlevel 57 goto EXIT
rem Now test for key value for 8 key (ie, the EIGHTH option)
if errorlevel 56 goto EIGHTH
rem Now test for key value for 7 key (ie, the SEVENTH option)
if errorlevel 55 goto SEVENTH
rem Now test for key value for 6 key (ie, the SIXTH option)
if errorlevel 54 goto SIXTH
rem Now test for key value for 5 key (ie, the FIFTH option)
if errorlevel 53 goto FIFTH
rem Now test for key value for 4 key (ie, the FOURTH option)
if errorlevel 52 goto FOURTH
rem Now test for key value for 3 key (ie, the THIRD option)
if errorlevel 51 goto THIRD
rem Now test for key value for 2 key (ie, the SECOND option)
if errorlevel 50 goto SECOND
rem Now test for key value for 1 key (ie, the FIRST option)
if errorlevel 49 goto FIRST
rem Check for value of 48 or greater (get another number)
if errorlevel 48 goto GET2
:FIRST
cd\wp
edit
goto EXIT
:SECOND
cd\spread
spread
goto EXIT
:THIRD
cd\mainfile
edit
goto EXIT
```

```
:FOURTH
cd\database
data1
goto EXIT
:FIFTH
cd\program
prog
goto EXIT
:SIXTH
cd\mouse
mouse
goto EXIT
:SEVENTH
cd\dos
goto EXIT
:EIGHTH
cd\service
edit svguide.doc
:EXIT
```

Using AUTOEXEC.BAT

This AUTOEXEC.BAT file provides a sample of the kind of startup procedures that a user might carry out when starting a computer work session.

The program first sets the PATH and APPEND command path options (refer to Chapter 3, 5, and 6 for more information about these commands). Then it establishes a sophisticated PROMPT that creates a more useful command-line prompt and then places the current date and time in the upper right area of the screen. Refer to Chapter 5, "Tips for Advanced Users" for more information on using the PROMPT command.

Next, the program displays

Skip options (Y/N)?

Then the GET.COM program is called into action. It provides a way to skip the menu of choices and therefore abbreviate the start-up procedure. The program will not continue until the user presses either the **Y** or **N** key.

If the user chooses **N** (for NO), the program ends. If the user presses **Y** (for YES) the menu of nine options is displayed (this batch file, "MENU.BAT," was discussed earlier in this chapter).

The next section of the batch program also uses the GET.COM program. When the user responds by choosing a number, a series of IF tests are used to conditionally transfer processing to a corresponding label where one or more programs are activated.

Creating Additional Batch Files

The batch file examples presented in this chapter can be entered and used as they are; but they are even more useful when modified to serve your special needs. You should think of these examples (and the examples presented in the next chapter) as only the starting points in your batch file arsenal: every time you find yourself repeating one or more DOS tasks fairly often, think of a way to create a batch file to carry out those tasks. Place the most useful batch files in one directory and then specify that directory using a PATH command on one line in your AUTOEXEC.BAT file. That way, you will be able to call those batch files into action while working in any directory on any drive.

Chapter 5
Tips for Advanced Users

This chapter is for any computer user who wants to know more about the inner workings of DOS: it provides advice and information about advanced DOS methods and presents examples you can try out on your computer system.

The topics selected for inclusion in this chapter are presented not only because they are of interest to advanced users, but also because they demonstrate some of the less well-known capabilities of DOS. We hope, therefore, that this chapter will prove interesting for skilled DOS users, and enlightening to less experienced users.

Advanced use of many of the DOS commands is described in this chapter, with special sections to describe the MODE, PROMPT, SET, and XCOPY commands. Also included is a section that describes the use of the CONFIG.SYS file and the related configuration options:

Some of the other topics described are:

Using ASCII characters
Using the DOS editing keys
Device names
Filter commands
Redirection and piping
DOS printing methods
Using laser printers
Using the GRAPHICS command

There are a large number of examples presented in this chapter. If in doubt, turn to these examples for help: each example is representative of typical use and each has been tested. The examples work: you can type them in and use them just as they are. They demonstrate some of the useful things DOS can do and can provide the starting point from which you can design your own set of advanced techniques.

NOTE:

Some of the examples presented in this chapter are complex; they often require at least a basic knowledge of the functions of DOS commands. If you have not done so already, we suggest that you read Chapters 1 through 4 of this book before trying these examples. If you are not completely familiar with a DOS command used in an example, you can find out more about it by referring to the related section in Chapter 6, "DOS Command Reference."

The Advanced DOS User

After using DOS for a while, you may find that there are some computer tasks that seem repetitive and inefficient. If you spend some time experimenting, you will start to discover some very interesting things about DOS (things not mentioned in the user's guides). When you start this kind of curious searching for new DOS solutions, you can consider yourself an "advanced" user (often referred to as a "power" user). Advanced DOS users are looking for the DOS tricks, the hidden secrets of the system. They have the kind of curiosity that leads to long hours at the computer keyboard trying to solve that "impossible" DOS problem (a power user never believes in the impossible).

I hope the techniques described in this chapter will be useful to that sort of advanced user. But I also hope that they will arouse that same kind of curiosity in the less-experienced user, that they will help demonstrate the depth and capability of this disk operating system.

Using ASCII Characters

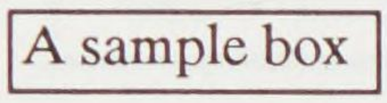
Some of the examples in this chapter require the use of special characters that are not related to any one of the keys on the keyboard. Nevertheless, these characters have an ASCII (American Standard Code for Information Interchange) value and can be produced from the keyboard. To produce one of these characters, you press and hold the **Alt** key while entering the ASCII value using the numeric keypad number keys (at the right side of the keyboard). Do not release the **Alt** key until you have entered the entire number; then, when you release the **Alt** key the corresponding character will be produced on the screen.

A number of graphics characters can be produced using this method. For example, try pressing and holding the **Alt** key while typing in the number 176. When you

release the **Alt** key, the following graphic character should be produced at the cursor position:

To see some examples of other graphics characters, repeat the same technique for characters numbered 177, 178, 179.

You can use this technique to draw a box. For example, use characters numbered 218, 196, 191, 179, 217, and 192 to surround text with such a box. Like this:



Appendix C lists the available ASCII characters.

Some ASCII values do not display a character. For example, ASCII value 007 returns a tone from the computer's speaker (beep). ASCII value 009 returns the TAB character. You can create files to store these characters by using the COPY CON option.

For example, to store the "beep" character in a batch file, enter

```
copy con beep.bat
echo off
```

Press the **Enter** key after entering each line.

Next, enter "echo" and a space. Then press and hold the **Alt** key and then press the number **7** key on the keypad. When you release the **Alt** key, the "^G" character will appear on the screen. Press the **Enter** key again. Finally, press and hold the the **Ctrl** key and then press the letter **Z** key. What you have entered so far should look like this:

```
copy con beep.bat
echo off
echo ^G
^Z
```

When you press the **Enter** key, the file will be saved in the current directory.

You can now produce the speaker's sound whenever you enter the batch filename, "BEEP."

NOTE:

Depending on the word processing program you are using, you may or may not be able to carry out this technique while you are creating a text file. Even if you can create the needed character on the screen, your word processing program may not have the capability of storing these characters in a file. If you have problems, try creating one or more examples of the characters you need using the COPY CON method described above; then you can use your word processor's capability to insert that file into your working file.

If your word processor does not properly save these special characters, read your program's manual to learn if you can save the file using an "ASCII-save" option.

Using the DOS Editing Keys

As soon as DOS is loaded, several editing keys become active. These keys can be used to edit your entries on the command line (when you enter characters next to the DOS prompt).

The editing keys work hand-in-hand with a special DOS capability referred to as the keystroke input buffer. When you enter characters on the command line and press the **Enter** key, DOS retains a record of the characters you entered. The editing keys provide an easy way to recall (and, if you want, modify) these characters the next time you are making another command line entry. The record of keystrokes held in the input buffer is sometimes referred to as the "template" of previous keystrokes.

By using the editing keys, you can quickly repeat all or part of the last command line you used.

The sections below describe each of the editing keys that are active while using DOS.

F1 Copies one character at a time from the input buffer to the command line each time this key is pressed. (You can also use the **Right Arrow** key.)

F2 Copies all characters from the input buffer to the command line up to the first instance of the character specified (entered after this key is pressed).

F3 Copies all characters from the input buffer to the command line.

F4 Skips over (does not copy) characters from the input buffer to the command line up to the character specified (entered after this key is pressed).

F5 Cancels all characters currently in the input buffer.

F6 Inserts an end-of-file character (^Z) in the command line.

Ins Inserts any characters entered into the input buffer from the position where this key is pressed.

Del Deletes a character from the input buffer at the position where this key is pressed.

Esc Abandons any characters entered and empties the input buffer.

Examples

The editing keys make re-using a command, or a slightly different version of a command, very easy. For example, if you enter

```
copy  letter1.txt  b:oldlet1.txt
```

and press the **Enter** key, the command is carried out AND the entire line is copied to the input buffer.

Now, if you press either the **F1** key or the **Right Arrow** key once, the first character of the previous line (in this case, the letter "c") will appear on the command line. Press the key again and the next letter ("o") will appear. You can continue to press the key until the entire line is in place. This key is often used when you want to re-use only part of the previous line. For example, if you want to copy the same file "LETTER1.TXT" to a different drive, you can use either the **F1** key or the **Right Arrow** key instead of re-entering the first part of the command. An easy way is to hold down the key until the portion of the message you want to use is on the screen. For this example, you can hold down the key until the copy command and the name of the file is on the screen:

```
copy  letter1.txt
```

Then you can enter the drive you want to send the copy to, resulting in

```
copy  letter1.txt  a:
```

You can also change a character or characters and then continue using this editing key to reproduce characters from the buffer. For example, if you want to change only the filename extension of the file to be copied, you could press the key until the characters

```
copy letter1.
```

appear on the screen. You could then enter the new extension and then continue to press the **Right Arrow** key to recreate the command line, resulting in

```
copy letter1.doc b:oldlet1.txt
```

The **F2** key has a slightly more complicated function. Using the same example, if you wanted to copy the same file but change the target destination, you can press the **F2** key and then the letter **b** key. The characters in the input buffer up to (but not including) the first occurrence of that letter will be copied to the command line resulting in

```
copy letter1.doc
```

You can then enter the new path designation.

The **F3** key is probably the most frequently used of all the editing keys. You can use it to recreate the command line using all of the characters from the input buffer.

The **F4** key is used to copy from the input buffer, but skip all characters up to the one you specify. Using the same example, if you want to call only the destination filename back to the command line, you can press the **F4** key followed by the letter **b** key. This will result in

```
b:oldlet1.txt
```

being copied to the command line.

The **F5** key is used to cancel the characters currently stored in the input buffer. Press the **F5** key and then press the **Enter** key to clear the buffer.

The **F6** key can be used to insert the end-of-file marker (^Z). It can be used to insert the marker into a command line or into a file when you are using the COPY CON option command to create a file.

The **Ins** key is used to insert characters when you are copying from the input buffer to the command line. Using the same example from above, if you wanted to insert a drive letter in front of the first filename, you can press the **Right Arrow** key five times to produce

```
copy
```

on the command line. If you now press the **Ins** key, you can enter the drive designator without overwriting characters in the input buffer. The command line will look like this:

```
copy  a:
```

Finally, you can press the **F3** key to copy the rest of the characters from the input buffer resulting in

```
copy  a:letter1.txt  b:oldlet1.txt
```

The **Del** key is used to delete characters when you are copying from the input buffer to the command line. We can reverse the insert process described in the previous example by again pressing the **Right Arrow** key five times to produce

```
copy
```

on the command line. If you now press the **Del** key twice, the next two characters in the input buffer will be deleted (in this case, the letters "a:"). If you now press the **F3** key to copy the rest of the characters from the input buffer, the result will be

```
copy  letter1.txt  b:oldlet1.txt
```

The **Esc** key can be used to cancel a partially-created command line. It also erases all characters from the input buffer.

DOS Device Names

Some of the examples in this chapter refer to standard DOS devices. The following list summarizes the DOS device names. These names are reserved for reference to devices and cannot be used as filenames.

CON	The combination of the keyboard console and the screen.
AUX or **COM#**	The serial port. Your computer can access one or more serial ports by number. Unless you use the form "AUX" when referring to a serial port, the port number (#) is always required.

PRN or **LPT#**	The parallel printer. The computer can access one or more parallel printers by number. Unless you use the form "PRN" when referring to a parallel port, the port number (#) is always required.

DOS Filter Commands

A filter is a command that can read input, transform it, and then output it (to the screen, to a file, or to the printer). Special DOS processing symbols are used with filter commands to re-direct the output.

DOS provides the following three filters:

- **FIND** Searches for one or more characters in a file.
- **MORE** Sends output to the display monitor, one screen at a time.
- **SORT** Alphabetically sorts text.

All three are text-oriented filters. They receive text, process it in some way, and then output it.

There are several examples in the following sections that demonstrate how these filters are used.

DOS Redirection and Piping

One of the most powerful, and yet seldom used capabilities of DOS, is its ability to redirect the input and output resulting from the use of DOS commands. Generally, DOS directs input from the keyboard to the screen: as you type in characters from the command line, they are displayed, but no other action is taken (until you press the **Enter** key). Then, when you press the **Enter** key, DOS looks at the characters you have entered and compares them with program names (stored files with the ".COM," ".EXE," or ".BAT" filename extension) found on the specified path. If a match is found, DOS assumes it is an executable program or batch file and attempts to run it immediately.

But DOS also looks for other special characters on the command line. These special characters tell DOS to take special actions with regard to the program names entered. These special characters can indicate redirection and piping.

Redirecting DOS Input and Output

These three symbols can be used to redirect both input and output related to DOS commands:

>	Used to direct the output of a DOS command. Any output that results from using a DOS command is sent to the destination specified after this symbol.
<	Used to direct input into a DOS command. Input is taken from the source specified after this symbol.
> >	Used to direct the output of a DOS command to the end of a file, if the specified file exists. Any output that results from using a DOS command is appended to the end of an existing file. If the file does not exist, it is created.

Redirection Examples

The > symbol can be used to direct the output of a DOS command to a destination other than to the screen. For example, you can use the COPY command to send the copied file to the printer. To send a copy of the file "NEWSALE.TXT" to the printer, enter

```
copy newsale.txt >prn
```

The file will be printed using only standard DOS formatting (certain special formatting characters inserted into the file by word processors are ignored).

Another useful version of redirection to the printer provides a printed copy of whatever output a DOS command will normally send to the screen. For example, you can use the DIR command and have the result sent to the printer.

To produce a printed directory listing of the the files on the "SALES" directory on the disk in drive A, enter

```
dir a:\sales >prn
```

When the directory listing is directed to the printer, it will not be displayed on the screen.

You can also use redirection to send output to a file. We can modify the example above to send the result of the directory listing to the file "DIRSAVE.TXT" by entering

```
dir  a:\sales  >dirsave.txt
```

If the object of the redirection is not a reserved device name, DOS will assume it is to be a file. If the file does not exist, it will be created. If the file already exists, the contents of the file will be replaced (overwritten).

You can also use redirection in a batch file. For example, the ECHO commands can be used to send text to a file. Enter the following line in your batch file:

```
echo  DISK1 - TEMPORARY BACKUP FILE >disk1.txt
```

When the batch file is run, the file "DISK1.TXT" will be created (if the file already exists, it will be overwritten) and the text "DISK1 - TEMPORARY BACKUP FILE" will be placed in it.

The symbol **<** can be used to direct input FROM a file to be used with a DOS command. One of the most common uses of directed input is with the SORT command. To alphabetically SORT an existing file and to direct the output to a new file, you use two redirection symbols (**>** for output and **<** for input). To direct input to the SORT command from the file "ADDRES3.TXT" and to direct the output to the file "LABEL3.ADD," enter

```
sort  <addres3.txt  >label3.add
```

Each of the above examples demonstrates redirection to create a file (or to overwrite an existing one). The symbol **> >** is used to direct output to an existing file without overwriting it. For example, to direct the output of the TREE command to the previously created file "C-FILES.TRE," enter

```
tree  c:  >>c-files.tre
```

The TREE command would normally display the names of all the directories on drive C on the screen. This command tells DOS to append the directory names to the end of the file "C-FILES.TRE" on the current directory.

By using a combination of input and output redirection, we can use a batch file to carry out the redirection tasks and also to create the labels for the file. For example, the following batch file can be used to create two different files and then to append labels and sorted output to them:

LABEL1.BAT

```
echo  off
echo  LABEL4.ADD - Sorted on column 12 (addresses) >a:label4.add
echo . >>label4.add
sort  /+12  <addres3.txt  >>a:label4.add
echo  LABEL5.ADD - Sorted on column 1 (names) >a:label5.add
echo . >>label5.add
sort  <addres3.txt  >>a:label5.add
```

This batch file first uses the ECHO command (in the second line) to send a line of text to the file "LABEL4.ADD" on drive A. If the file does not exist on the root directory of drive A, it is created; if the file does exist, it is overwritten (existing text in the file will be lost).

The third line directs the output of the ECHO command (a period) to the same file indicated in line two (LABEL4.ADD), but this time the symbol > > is used to APPEND the result to the file rather than overwriting it. This line demonstrates that the ECHO command can be used to add "formatting" lines to an existing file. The "blank" line will separate the title text (created in line two) from the sorted output (that will be created by the next line).

NOTE:

I have placed a period on this ECHO line because some versions of DOS will not pass through a blank line to the target file (even if you enter spaces). And, if DOS encounters a blank line, the ECHO command is seen as a request for the ECHO status. As a result, the message "ECHO is off" would have been appended to the named file. It is also possible to use an ECHO command to send (or append) the ASCII value for a line feed or carriage return to the target file (refer to the section, "Using ASCII Characters" in this chapter). However, this technique may also cause problems, depending on the version of DOS you are using and on the capabilities of your word processor.

The fourth line uses the SORT command to alphabetically sort the file "ADDRES3.TXT" (sorting on the character in column 12). The resulting sorted text is also directed to the "LABEL4.ADD." It is added to the file on the next line after the "formatting" line (the line with the period on it).

The fifth line is used to again send a line of text to a new file "LABEL5.ADD." As before, if the file does not exist on the root directory of drive A, it is created; if the file does exist, it is overwritten.

The sixth line is another formatting line and the last line uses the SORT command to again alphabetically sort the file "ADDRES3.TXT." But this time, the command sorts on the character in column 1 and the resulting sorted text is directed to (appended to) the file "LABEL5.ADD."

When you run this batch file, the result is two new sorted files, each different, but both formatted clearly with an appropriate title as the first line.

When we run this batch file, only the message "echo off" (from the ECHO command on the first line) will be displayed. All other output from the program is directed to files.

READ.BAT

A very useful little batch file, "READ.BAT," takes advantage of the capabilities of the MORE filter command to provide a way to quickly read any file on any disk.

READ.BAT

```
echo off
\dos\more < %1
```

In this example, the directory path "\DOS" is the directory where your DOS external programs are stored. The MORE filter command is used with redirection to display the file that is specified on the command line. You can specify any pathname at the same time you enter the filename.

For example, if you enter

```
read b:\letters\newsale.txt
```

the file "NEWSALE.TXT" on drive B in the "\LETTERS" directory will be displayed, one screen of text at a time.

Redirection to NUL

NUL is a reserved device name that is not really a device at all. It is actually a handy DOS feature that can be thought of as a "dummy" device that you can specify when

you really don't want the specified output to go to a device at all; usually you use NUL when you want the output directed to "nowhere" (anywhere OTHER than to a device). Usually it is used to direct output that you do not want to appear on the screen. This is sometimes called redirecting output to the "bit-bucket" or to outer space. The fact is, it is merely a signal to DOS to redirect the output, but not to send it to one of the standard devices.

> **NOTE:**
> If you specify NUL as the input device, an end-of-file marker is generated.

The use of redirection with NUL is a DOS feature that can often be very useful. We have created another one of those "quickie" batch files that makes use of the FIND filter in a batch program we have named "PAUSE1.BAT." It is used to create a timed pause. The FIND filter is used because it does not write to a file (it doesn't make any changes in it); however, when used to find a known character in a known file, we can accurately determine the amount of time it will take to execute. If you direct the output of the FIND command to NUL, no output will be sent to the screen; the user of the batch file will see it as a pause in processing. The PAUSE file can be created with two lines (but you can repeat the FIND line to create longer pauses).

PAUSE1.BAT

```
echo off
find  "a"  c:\dos\command.com >nul
```

The time it takes DOS to find the letter "a" in the specified file, will appear as a pause in the batch file. To create longer pauses, repeat the line.

Instead of using COMMAND.COM as the search file you can create a special "time-waster" file with nothing in it but one letter or number repeated the number of times corresponding to the timed pause you want to create. To create a variable timer, create a file with each character repeated a specified number of times. Then you can search for the letter that corresponds to the time-out you want. You can determine the length of the pause by entering a parameter on the command line that can be used within the file. The PAUSES.BAT file below uses that method.

PAUSES.BAT

```
echo off
find  "%1"   \dos\pause-1.txt >nul
```

The file "PAUSE-1.TXT" should contain something like the following:

PAUSE-1.TXT

```
1111111111 1111111111 1111111111 1111111111 1111111111
2222222222 2222222222 2222222222 2222222222 2222222222
2222222222 2222222222 2222222222 2222222222 2222222222
3333333333 3333333333 3333333333 3333333333 3333333333
3333333333 3333333333 3333333333 3333333333 3333333333
3333333333 3333333333 3333333333 3333333333 3333333333
4444444444 4444444444 4444444444 4444444444 4444444444
4444444444 4444444444 4444444444 4444444444 4444444444
4444444444 4444444444 4444444444 4444444444 4444444444
4444444444 4444444444 4444444444 4444444444 4444444444
```

The file "PAUSE-1.TXT" contains only numbers. Notice that each larger number appears correspondingly more often than the number of ones. This should result in correspondingly longer pauses for each higher number you search for. If you enter

pauses 3

it will create a pause three times the duration of

pauses 1

DOS Command Piping

DOS command piping is similar to redirection. It provides a way to use the output of one command as the input for another. DOS piping automatically redirects the output and input when you use the command piping symbol (¦).

Piping Examples

A common use of piping is to pipe output to a filter command. For example, the DIR command, when used to display the files in a directory with many files, will rapidly display more than one screen full of file listings; the files will scroll off the screen faster than you can read them.

You can view a directory one screen at a time by entering

dir ¦ more

This will pipe the output of the DIR command to the MORE filter. The listing will be displayed one screen at a time; you can press any key to view the next screen full of data or press and hold the **Ctrl** key and then press the **Break** key to stop the display.

DIROF.BAT

The simple piping method described above can be used in a handy batch file we have named "DIROF.BAT." It provides a quick and easy way to get a one-screen-at-a-time directory listing of any directory on any drive. Enter:

DIROF.BAT

```
echo  off
echo  Ready to display directory [ %1 ]
echo  Press any key to view the next screen full of data.
dir  %1  ¦  more
```

When you enter the name of this batch file without adding a path designation, the program will display the directory listings in the current directory. Enter

dirof

and all the files in the current directory will be displayed, one screen at a time.

If you want DOS to display the contents of the "WORDS" directory on drive C, you can pass the entire path designation to the batch file as a parameter. Enter

```
dirof  c:\words
```

The batch program will display

```
Ready to display directory [ c:\words ].
```

and then the specified directory will be displayed, one screen at a time.

DSORT.BAT

You can also pipe more than one command output at a time. The following batch file is similar to the "DIROF.BAT" program above. The file "DSORT.BAT" sorts the directory listing alphabetically before displaying it one screen at a time.

DSORT.BAT

```
echo  off
echo  Ready to display directory [ %1 ]
echo  Press any key to view the next screen full of data.
dir  %1 ¦ sort ¦ more
```

DBUT.BAT

You can also use piping to combine commands, resulting in more powerful versions of the standard commands. This example, "DBUT.BAT," displays a directory of all files EXCEPT the file specification you enter.

DBUT.BAT

```
echo  off
dir ¦ find /v "%1"
```

The /V option of the FIND filter command, displays all lines not containing the specified characters. When used with the DIR command, directories that match the files you specified are not displayed. Before the directory is displayed, the characters you enter after the batch filename (on the command line) will be searched for by the FIND command.

For example, if you enter

```
dbut .COM
```

all the files in the current directory will be displayed, EXCEPT the files that include the ".COM" filename extension. Don't forget to enter filename characters in upper-case, because that's the way they appear on the display lines. If a match is found in any directory line, the entire line will not be displayed.

Since the FIND command searches specifically for character matches, the wildcards will not work with this batch file.

You can also use piping with more than two commands. For example, you can add the MORE filter command to the above batch file.

```
echo off
dir | find /v "%1" | more
```

This version of the batch file acts just like the original, except that the output is piped to the MORE command; as a result, only one screen of data is displayed until you press a key to resume displaying.

You can use this command to search for other characters on the displayed lines besides the file names. For example, if you enter

```
dbut -86
```

file listings that include the "-86" character string will not be listed; in other words, the line for files that were created in 1986 will not be displayed.

DLOOK.BAT

To use the same capability as in "DBUT.BAT" for selective display of files that include characters that ARE found on the display line, you can create the file "DLOOK.BAT."

DLOOK.BAT

```
echo  off
dir  ¦  find  "%1"  ¦  more
```

This small batch file uses the FIND command to display directory lines that contain the characters found. To display only the files created in 1986, enter

dlook -86

Since all the directory lines you are searching for will end with the date (separated by hyphens) the lines that include "-86" will be found and displayed.

Other Redirection Techniques

The examples in this section are only a few of the many useful techniques you can carry out using redirection and piping. The best methods result from studying your needs and then thinking through the DOS commands that might result in a solution. A good set of custom batch files created using these techniques will result in much more efficient use of your computer. It's a sure sign of a DOS power user.

Keep these batch files in one directory that you specify with a PATH command in your AUTOEXEC.BAT file. That way, you can use one of these batch programs while working in any directory.

Configuring Your System

DOS provides a set of configuration commands that can be used to set up your special DOS needs. These commands must be stored in the configuration file, "CONFIG.SYS."

CONFIG.SYS

CONFIG.SYS is not a DOS command; it is a special configuration file used by DOS. Each time your computer is started (or reset) DOS checks to see if there is a CONFIG.SYS file stored on the root directory of the startup disk. Each line of this file must be a recognizable DOS configuration option (described in this section).

CONFIG.SYS is a file, not a program. You can create the file with any word processing program or using the COPY CON option.

DOS looks first to this file to see if it contains special information related to devices used by your computer and other special applications. It may contain information about your monitor, your printer, or other devices or functions.

Many programs require a special CONFIG.SYS file to be placed on the startup disk. These programs will not operate properly without this file available to DOS upon startup.

Often, this file will contain the single line:

DEVICE=ANSI.SYS

This line must be stored in a CONFIG.SYS file in order for certain special DOS functions to operate. ANSI.SYS is the standard device driver for keyboard control.

You can quickly create a CONFIG.SYS file that contains this single line by using the COPY CON option (described earlier in this chapter in the "Using ASCII Characters" section).

The CONFIG.SYS file may also contain information DOS needs to set up special memory allocations for reading and writing files or other special configuration settings. The file may contain lines that relate to one or more of the DOS configuration options described in the following sections.

BREAK

Format:

BREAK=(ON or OFF)

The BREAK option is used to carry out extra checks to determine if the **Ctrl** + **C** or **Ctrl** + **Break** key key combination has been pressed.

If you do not use this option in the CONFIG.SYS file, BREAK is set to OFF. DOS will still check to determine if the **Ctrl** + **C** or **Ctrl** + **Break** key combination has been pressed, but only during input from the keyboard, output to the printer, output to the screen, or output to the serial port. In other words, when the BREAK option is OFF, pressing this key combination will not interrupt disk operations; this protective feature avoids the file storage problems that can result from interrupting disk input and output operations.

If BREAK is set to ON, you can exit from many programs even while they are executing. Some application programs may assign the **Ctrl** + **C** or **Ctrl** + **Break** key combination to carry out special functions. Setting BREAK to ON may disrupt the operation of such programs.

Example

To set break to ON, enter the following line in your CONFIG.SYS file:

break=on

BUFFERS

Format:

BUFFERS=n

The BUFFERS command is used to tell DOS how many disk buffers (n) are to be maintained. Each buffer set reserves more system memory (RAM). The buffers you set up are used to hold the most recently used data. That way, the next time the computer needs the data it will already be in memory and processing will be speeded up. Buffers are also used to compensate for the fact that different devices operate at different speeds; the extra buffers can be used by DOS to temporarily hold data being sent to devices.

Each buffer reserved uses 528 bytes (512 plus 16 bytes used by the system). If you do not use this option in the CONFIG.SYS file, two buffers are automatically set by DOS if you have the minimum PC configuration. Three buffers are the default if your system includes a high-density disk drive of any type. However, if your system includes memory beyond the minimum 256K-bytes, the buffers are automatically set to ten. If your system has more than 512K-bytes, the buffers are automatically set to 15. If your system includes this extra memory, you cannot override the buffers setting by entering a number lower than the default.

Typically, using this option to set aside extra buffers (if you have the extra memory available), will increase your computer's performance. Setting up more than ten buffers will often speed up your computer's file handling capabilities. You can try different settings to see if you notice any difference in performance.

Example

To set buffers to 20, enter the following line in your CONFIG.SYS file:

```
buffers=20
```

COUNTRY

Format:

```
COUNTRY=country code,[code page][,][d:][filename]
```

The COUNTRY option is used to specify country-specific information such as the date, time, collating sequence and folding format, and capitalization for a country.

The country code is a three-digit international code (based on the telephone system codes). For more information on country codes, refer to Appendix B.

The code page contains the country-specific information for the specified country. Each country available to the system should have two code pages already set up. The code page number entered here specifies which of the two will be used.

The filename you specify should contain information related to the selected country.

If you do not use this option in the CONFIG.SYS file, the country code is set to 001, the code page is set to 437, and the country information file is set to COUNTRY.SYS. These are the settings for the United States.

If you omit the country information file when using this option, DOS assumes that the file COUNTRY.SYS will be found in the root directory of the startup.

Example

To set the country code 001, leave the code page as the default (437), and use the standard information file (to be found in the "\DOS" directory) enter the following line in your CONFIG.SYS file:

```
country=001,,c:\dos\country.sys
```

DEVICE

Format:

DEVICE=(driver name)

The DEVICE option is used to tell DOS which device driver software to use for devices. DOS loads standard device drivers that support standard input and output, the clock device, the standard printer, and the disk drives. However, special devices, or special handling of devices may require some form of this command in the CONFIG.SYS file.

Many application programs will require you to place specific device driver commands in this file. As stated above, some special operations of DOS related to keyboard input will not operate unless the line "DEVICE = ANSI.SYS" is included in this file.

Example

To use the file "ANSI.SYS" as the standard device driver, enter the following line in your CONFIG.SYS file:

```
device=ansi.sys
```

FCBS

Format: FCBS=n1,n2

The FCBS option is used to specify the number of file-control blocks for file sharing that can be opened by DOS.

The first number (n1) specifies the number of files opened by this command that can be open at one time. This value can be any number between 1 and 255. If this number is not specified, it is set to 4.

The second number (n2) specifies the number of files opened by this command that cannot be closed, even if DOS needs the file space for other programs. This value can be any number between 1 and 255, however, it must be equal to or greater than n1. If this number is not specified, it is set to zero.

When file sharing is in operation, the number of files that can be open at the same time may be limited by this command. When file sharing is not in operation, this command is not applicable.

Example

To set the open files to 6 and 10, enter the following line in your CONFIG.SYS file:

```
fcbs=6,10
```

FILES

Format:

```
FILES=n
```

The FILES option is used to specify the maximum number of files that can be open at the same time. DOS reserves an area of memory to keep track of each file opened.

This number (n) can be any value between 8 and 255. If you do not specify this option in your CONFIG.SYS file, FILES will be set to 8.

Setting the FILES option to a higher number uses more system memory, but may speed up processing. When setting this number you should be aware that three files are open any time DOS is operating (COMMAND.COM and two hidden system files). Try setting this option to various values to determine which is best for your applications.

Example

To set the maximum number of files that can be open at one time to 15, enter the following line in your CONFIG.SYS file:

```
files=15
```

LASTDRIVE

Format:

LASTDRIVE=drive letter

The LASTDRIVE option is used to set the maximum number of drives that can be accessed.

drive letter is a letter from A to Z used to specify the last (highest) drive designator letter than can be accessed by the system. For example, if this value is set to D, only drive designations A through D can be accessed. Set this to a letter greater than the last physical drive designation letter if you plan to access virtual (non-physical) drives (for example, if you plan to use the SUBST command).

If you do not use this option in the CONFIG.SYS file, the LASTDRIVE is set to E (the system can access drives A-E).

Example

To set the last drive designation letter to Z, enter the following line in your CONFIG.SYS file:

lastdrive=z

SHELL

Format:

SHELL=[d:][path]filename

If your computer is to use any command processor other than the standard COMMAND.COM program, the SHELL command must be used in the CONFIG.SYS file. If DOS finds this line in the CONFIG.SYS file, the named file will be loaded at startup (or restart) instead of the COMMAND.COM program.

Example

If you are using a modified command processor named "COMMAND2.COM, enter

shell=command2.com

When your computer is re-started, DOS will load this file instead of COMMAND.COM.

STACKS

Format:

STACKS=number,size

The STACKS option is used to set the the number of stack frames and the size of each stack frame. Increased memory (RAM) is reserved for each higher number you set for this option,

When a hardware interrupt occurs, DOS uses one frame from the stack. After the interrupt has been acted on, the stack frame is returned to the stack. Limited stack resources can result in stack errors; increase the stack resources by setting higher values for one or both of the STACKS options.

number is the number of stack frames available to the system. This option can be set from 8 to 64. If you do not set this option, it will be set to a default value depending on the design of your system. If stack errors occur, try increasing this value.

size is the memory size, in bytes, of each stack frame. This option can be set from 32 to 512. If you do not set this option it will be set to a default value depending on the design of your system.

Example

To specify that stack support should not be installed (no dynamic stacks are available), enter the following line in your CONFIG.SYS file:

stacks=0,0

To set the number of stack frames to 18 and the size to 128, enter

stacks=18,128

DOS Printing Methods

Most DOS users have a printer attached to their personal computer. However, they mostly send output to that printer via the printer drivers that are part of their application programs. Nevertheless, with some knowledge of how DOS sends signals to the printer, you can gain a measure of control over the printer without going through an application driver.

You can use features that are built into DOS to change the type of output produced by your printer. One way is to use the MODE command (described later in this chapter) to change the default characteristics of the printout. For example, you can change the maximum number of characters per line your printer will output (across the page) or the number of lines your printer will print per inch (down the page).

Once you have reset these features, they will stay set that way until you again use the MODE command to change them, or until you load a different printing program that sends new signals to the printer (loading your word processor may change the printer output default settings). Probably the easiest way to reset the printer output parameters back to the default settings is to turn the printer off and back on again. This should be done using the printer's main power switch (the online/offline switch will not reset these configuration options).

The techniques described in the following sections are generally useful if you have a parallel printer attached to your computer via the parallel printer port. You can also use a serial type printer with DOS, but you generally will have to redirect the computer's output to the serial port (where the printer is attached) by using the MODE command. For more information, refer to the MODE command section in this chapter.

PRINT Program Options

Once you have reset your printer's configuration options, they will be in effect when you print using the DOS PRINT command. To review the PRINT options as described in Chapter 2, you enter the PRINT command followed by the filename you want to print. For example, if you want to print the file "LETTER2.TXT" that is stored in the "SALES" directory on the diskette in drive B, enter

```
print  b:\sales\letter2.txt
```

The advantage to using the PRINT command is that you can go back to work using other programs while the file is being printed.

You can queue up files (set up more than one file to be printed sequentially), a feature not offered by some word processing printer programs.

Wildcard characters are also allowed. For example, if you wanted to print all of the letters found in the "SALES" directory, you could enter

```
print  b:\sales\letter*.*
```

All of the files in the "SALES" directory on drive B that begin with the characters "LETTER" will be printed. The printing attributes set with the MODE command will still be in effect and you can still continue working while the series of files is printed.

Other Printing Methods

In addition to the DOS PRINT command, you can use other DOS commands to direct output to the printer. For more information on redirecting the output of DOS commands to the printer, refer to the section "DOS Redirection and Piping" in this chapter.

You can get a quick print of whatever is displayed on the computer's screen by pressing and holding the **Shift** key and then pressing the **PrtSc** key. If your printer is turned on, this key combination will immediately send a copy of the screen display to the printer.

You can also send output to the printer using the **PrtSc** key. If you press and hold the **Ctrl** key and then press the **PrtSc** key once, DOS will send any subsequent characters that are displayed on the screen to the printer. This "echoing" to the printer will stay in effect until you again press the **Ctrl** + **PrtSc** key combination. This is a quick way to print a copy of information produced in response to other DOS commands. For example, after you have used the **Ctrl** + **PrtSc** key combination, the command

```
dir  a:*.txt
```

will print out both to the screen and to the printer a list of all the files on the root directory of drive A that include the .TXT filename extension. You can print an entire file by displaying it using the TYPE command. When you are done printing, don't forget to again press the **Ctrl** + **PrtSc** key combination again.

Special Printer Capabilities

Some printers have special capabilities. For example, many printers can re-strike printed characters for a darker and bolder looking output. "Graphics" printers may also have the capability of varying the size of the printed characters. You can tell such printers to change character styles by sending to the printer a software-driven

signal known as the "escape code." The escape code prepares the printer to accept a new value which, in turn, acts on preprogrammed settings within the printer to change the style of characters that are subsequently printed.

Programming languages such as BASIC provide a way to send such signals to the printer. After such a signal is sent to the printer, the style of the output changes - depending on the way that particular printer interprets the signal.

For example, the BASIC command

```
LPRINT CHR$(15);
```

sends the escape code (CHR$) to the printer followed by the ASCII value 15. Many graphics printers will interpret this signal as a command to change to "condensed mode." In this mode, the printer will print much smaller characters. The printer will continue to print in condensed mode until turned off or reset.

With this in mind, it becomes clear that BASIC language commands can be used to create simple programs to change the printing output styles (if your printer "understands" these signals).

A Sample Printer Program in BASIC

You can use the BASIC programming language to create a program that offers a menu of choices of printer output styles. The sample program reproduced below is a simple BASIC program used to turn your computer into an electronic typewriter. It takes a user choice from a menu of choices and then uses the escape codes to change the printer's character output type.

TYPER.BAS

```
10 CLS
20 PRINT
30 PRINT " ************************"
40 PRINT "**      TYPER       **"
50 PRINT "**"
60 PRINT "**  Copyright 1987  **"
70 PRINT "************************"
80 PRINT
90 PRINT
100 PRINT "Choose a printing option (for Epson-compatible printers)"
110 PRINT
120 PRINT "    1  Near Letter Quality        8  Double Strike"
```

```
130 PRINT "    2  Pica                          9  Backspace"
140 PRINT "    3  Elite                         10 Subscript"
150 PRINT "    4  Superscript                   11  1/8 Inch Line Spacing"
160 PRINT "    5  1/6 Inch Line Spacing         12  Select Graphics"
170 PRINT "    6  7/72 Inch Line Spacing        13  Underline"
180 PRINT "    7  Enlarged                      14  Condensed"
190 PRINT
200 PRINT
202 LOCATE 22,8
204 COLOR 0,7
210 INPUT "Enter a selection number (1-14) and press the Enter key
",L: COLOR 7,0
220 IF L = 1 GOTO 410
230 IF L = 2 GOTO 430
240 IF L = 3 GOTO 450
250 IF L = 4 GOTO 470
260 IF L = 5 GOTO 490
270 IF L = 6 GOTO 510
280 IF L = 7 GOTO 530
290 IF L = 8 GOTO 550
300 IF L = 9 GOTO 570
310 IF L = 10 GOTO 590
320 IF L = 11 GOTO 610
330 IF L = 12 GOTO 630
340 IF L = 13 GOTO 650
350 IF L = 14 GOTO 670
360 PRINT
370 PRINT "Setting printer to option: "; L
380 PRINT
390 PRINT "
400 PRINT
410 LPRINT CHR$(27); "x"; CHR$(1);     ' 1   NLQ
420 GOTO 680
430 LPRINT CHR$(27); "P";       ' 2   Pica
440 GOTO 680
450 LPRINT CHR$(27); "M";       ' 3   Elite
460 GOTO 680
470 LPRINT CHR$(27); "S"; CHR$(0); CHR$(15) ' 4  Superscript
480 GOTO 680
490 LPRINT CHR$(27); "2";             ' 5   1/6 inch line spacing
500 GOTO 680
510 LPRINT CHR$(27); "1";             ' 6   7/72 line spacing
520 GOTO 680
530 LPRINT CHR$(27); "W"; CHR$(1);    ' 7  Enlarged
540 GOTO 680
550 LPRINT CHR$(27); "G";       ' 8  Double strike
560 GOTO 680
```

```
570 LPRINT CHR$(8); CHR$(8);      '9  Backspace
580 GOTO 680
590 LPRINT CHR$(27); "S"; CHR$(1); CHR$(15) '10  Subscript
600 GOTO 680
610 LPRINT CHR$(27); "0";       '11  1/8 Inch line spacing
620 GOTO 680
630 LPRINT CHR$(27); "x"; CHR$(1);     '12  Select Graphics
640 GOTO 680
650 LPRINT CHR$(27); "-"; CHR$(1);     '13  Underline
660 GOTO 680
670 LPRINT CHR$(15);       '14  Condensed
680 CLS
690 PRINT
700 PRINT
710 PRINT " ************************************************";
720 COLOR 0,7:PRINT " You have selected option number: "L:
730 COLOR 7,0
740 PRINT "************************************************"
750 PRINT
760 PRINT
770 PRINT " ************************************************"
780 PRINT "You can now use your computer as an electronic "
790 PRINT "typewriter. Your printed output will continue to use
800 PRINT "this setting until you turn off your printer or reset it."
810 PRINT "************************************************"
820 PRINT
830 PRINT
840 PRINT " ";
850 COLOR 0,7:PRINT " Press any key to clear the screen and con-
tinue. ":COLOR 7,0
850 TOS$=INKEY$:IF TOS$="" THEN 850
860 CLS
870 PRINT " 1   5    10   15   20   25   30   35   40   45   50   55   60   65
70   75"
880 INPUT "", X$
890 LPRINT X$
900 GOTO 880
```

You can enter the TYPER program just as it appears here and use it to turn your computer and printer into an electronic typewriter.

When you enter a number, a series of IF tests are used to transfer processing to one of the "LPRINT" lines (410-670) where the ESCAPE code and value are sent to the printer.

After the value is sent, the program displays a series of numbers across the top of the screen (to show spacing).

You can type in characters at any position on the screen. When you press the **Enter** key at the end of a line, the characters on that line will be immediately sent to the printer.

The printer will print characters in the selected style until you either turn the printer off or reset the printer by running the program again and selecting another style.

For example, if you choose option number 14 (condensed), the output will look similar to this:

This is a test of TYPER - condensed type

But if you choose option number 7 (enlarged), the output will look similar to this:

```
This is a test of TYPER - enlarged type
```

You can even exit the program if you want; the printer will still print in the selected style, as long as you do not switch off the printer's power switch or reset it.

A Batch File Printer Program

If you do not want to use a programming language like BASIC to send the escape code and reset values to the printer, there is also a way to do it using a DOS batch file. The program requires the use of GET.COM or a similar program. The creation of the GET.COM program was described in detail in Chapter 4, "Using Batch Files."

The example below recreates an abbreviated version of the menu choice structure of the TYPER.BAS program using DOS commands in a batch file. You can create the file using the DOS COPY command (COPY CON) or using any word processor that can store files in "standard" ASCII form.

TYPER.BAT

```
echo off
cls
echo   Print Options
echo
echo   1  DOUBLE STRIKE
echo   2  CONDENSED
echo   3  PICA
echo   4  ELITE
echo   5  EXIT TO DOS
echo
echo   Type in the number for your choice
echo
:GET1
get
rem Value higher than 54 is in error so get another number
if errorlevel 54 goto GET1
rem Now test for key value for 5 key (ie, the FIFTH option)
if errorlevel 53 goto FIFTH
rem Now test for key value for 4 key (ie, the FOURTH option)
if errorlevel 52 goto FOURTH
rem Now test for key value for 3 key (ie, the THIRD option)
if errorlevel 51 goto THIRD
rem Now test for key value for 2 key (ie, the SECOND option)
if errorlevel 50 goto SECOND
rem Now test for key value for 1 key (ie, the FIRST option)
if errorlevel 49 goto FIRST

rem Check for value of 48 or greater (get another number)
if errorlevel 48 goto GET1
:FIRST
echo (<ALT>155G)  >lpt1
goto EXIT
:SECOND
echo (<ALT>15)  >lpt1
goto EXIT
:THIRD
echo (<ALT>155P)  >lpt1
goto EXIT
:FOURTH
echo (<ALT>155M)  >lpt1
goto EXIT
:FIFTH
:EXIT
```

The construction of this batch file is essentially the same as that of the file "MENU.BAT" described in Chapter 4, "Using Batch Files." Refer to that chapter for more information on the function of such files. As suggested in that chapter, you can use the ASCII box-drawing characters to create a more attractive menu structure.

When you enter the name of this batch file, "TYPER," the menu of printer options will be displayed. The program displays the menu structure and then at the label "GET1" uses the GET.COM program to take a one-keystroke input from the keyboard (it passes the key's ASCII value to ERRORLEVEL).

Next, a series of IF tests checks the current setting of ERRORLEVEL to determine which key was pressed. If the user presses any key other than a number key between one and five, the program transfers processing to the GET1 label again where GET.COM waits for another keystroke.

If the user presses a number key between one and five, the processing is transferred to the corresponding label. Notice the special entry format required after each of the labels. For example, the line following the label ":FIRST" is

```
echo  (<ALT>155G)  >lpt1
```

This means that you enter the ASCII value by pressing and holding the **ALT** key while you enter the number using the keys on the numeric keypad (at the right side of the keyboard). When you use this method, DOS displays the ASCII character represented by that number. The ASCII values available to DOS and the characters the values represent are illustrated in Appendix C.

The reason we enter the character for value 155 (instead of the standard escape code of 27 as in the BASIC example) is that DOS receives the escape code value of 27 as an input from the ESC key and takes no action. We can "force" DOS to send the value to the printer by adding 128 to 27 yielding a value of 155 to be sent to the printer. This changes the high-order eighth bit in the character and it is sent to the printer. But, since the printer does not read the eighth bit, it receives the value as 27 and accepts the next value it receives. In this case, the value we want to send is the ASCII value for the capital letter "G." This is the value that - when received by the printer - resets the printer to double-strike mode.

In response to a user's selections, processing is transferred to a label where a corresponding value is sent to the printer.

NOTE

You may have trouble entering the ASCII codes with the **ALT** key (especially the lower values) because these values may be received by your word processing program as a control value used internally. In order to place the needed ASCII character in the batch file, you may have to create the display character for these values by using the DOS COPY command (use the COPY CON option) and then read the result into your word processing file.

Simple Printer Batch Files

If you commonly use only one form of special printing, you may not need as complicated a program as either of the above. For example, you may only need compressed printing (to print in 132 character mode). If so, you can create a small batch file just to set that mode of printing. The following batch file sets the condensed mode:

CONDENSE.BAT

```
echo off
echo (<ALT>015)  > lpt1
```

All you have to do is make sure the printer is turned on and then enter the name of the batch file. The batch file sends the escape code (via the redirected ECHO command) and value to the printer.

The printer will immediately be reset to the condensed mode. Using any of the DOS printing options will result in the small-sized character output.

Again, after the printer is set to the condensed mode, it will stay in that mode until you switch off the printer's power switch or reset it.

Printer Control Using PROMPT

Another way to send these escape characters is by using the PROMPT command. (For more information about this command, refer to the PROMPT command in the "Advanced DOS Commands" section later in this chapter.) Normally used to change the DOS prompt, this command also provides the ability to send an escape

sequence. This gives us a way to get around the limitation described in the previous section when DOS interprets the escape code as the **ESC** key and takes no action.

It is a technique that "sends" the escape code to the printer using the **Ctrl** + **PrtSc** key combination.

Just as the escape code was sent to the printer in the previous sections using a BASIC program or a batch program, you can also send the same code using the PROMPT command. For example, to set the Elite print mode using the PROMPT command, first press and hold the **Ctrl** key and then press the **PrtSc** key. This key combination sets up the printer to accept output from the keyboard. Anything you type in using the keyboard will still be displayed on the screen, but it will also be passed (echoed) to the printer (as soon as you press the **Enter** key).

If you enter

```
prompt $eM
```

and press the **Enter** key.

This will send the escape code followed by the letter "M" to the printer. When you enter this form of the PROMPT command, you will notice that the letter "M" and (usually) an error message is printed in the new Elite style. Press the **Ctrl** + **PrtSc** key combination again to turn off the echoing to the printer.

You will also notice that the DOS prompt has been reset to the letter "M." To change the prompt back to the one you were using, you will have to re-enter it. For example, if you want the prompt to display the current drive and the current directory followed by the right angle bracket (>), enter

```
prompt $p$g
```

Your printer outputs will still be printed in Elite character style. It will continue to print in the Elite mode until the printer is turned off or reset; this allows the possibility of some interesting printouts because it provides a way to combine printing modes. For example, if you reset the printer and then enter

```
prompt $eG
```

to set the double-strike mode, you can then enter

```
prompt $eM
```

Then the printer will print using the elite mode AND double-strike. Again, both of these printing effects will remain in effect until the printer is turned off or reset.

This method of using the PROMPT command to reset the printer mode assumes that two additional factors be in effect. First, your printer must be "standard" in the sense that it uses these particular escape sequences to reset the printing modes. If a printer uses different escape sequences you can still use this method, but you will enter the appropriate sequences described in the user's manual for that printer. Second, the PROMPT command will not work in this way if the ANSI.SYS file has been invoked in the CONFIG.SYS file. Check the contents of the CONFIG.SYS file (it contains normal ASCII text and can be displayed using the TYPE command or your word processor). If the line

```
DEVICE=ANSI.SYS
```

appears anywhere in that file, it must be removed before using the PROMPT command to send the escape code to the printer.

Using Laser Printers

DOS generally does not do an especially good job of outputting to laser printers. This is because you cannot program DOS to recognize special codes embedded in text files, the kind of codes that can signal output to a laser printer. DOS reads printed text from the specified file (or from the screen) and passes it along to the printer along with only a few basic formatting characters.

Many of the printer programs that come bundled with word processing programs (or with application programs) do not do such a great job of sending useful output to laser printers either. This is due to the fact that laser printouts are useful only to the degree that they can create text of varying sizes and styles using different font families (individual character designs). In order to print using this far more complicated information, the computer, the application program, and the laser printer must be closely matched.

To assure proper communication between computer and printer, modern application programs will ask for the type of printer during the installation phase. A laser printer must be "installed" before the program can use it. This is usually accomplished by making a special printer driver program available to the application program. The laser printer must be of the specific brand and type designated by the program if it is to decipher the complex signals output to it.

DOS, however, sends "standard" signals to the printer. If your laser printer can "emulate" standard dot-matrix printers, you can generally get a fairly clean looking output by using any of the DOS printing methods. However, the laser printer's capability to emulate standard printers may not extend to utilizing the escape code values described in the previous section.

Using the GRAPHICS Command

Graphics printers can also reproduce just about any character that can be displayed, but first you have to use the GRAPHICS command. Simply enter

graphics

and press the **Enter** key prior to attempting the graphics printout. The GRAPHICS command must be entered before you attempt to print out a graphics screen using the **Shift** + **PrtSc** key combination. Certain programs also require the GRAPHICS command to be in operation before they can print out graphics screens (this includes programs you write in BASIC).

The extended ASCII characters described in the previous section are another matter. Whether or not you are using a graphics printer, if is often difficult to get a printout of the box-drawing ASCII characters (described in the section "Using ASCII Characters" earlier in this chapter). Many printers simply cannot interpret these character values or they can't reproduce them. The extended ASCII characters will usually be printed, but as some other character. On the other hand, some laser printers and some low priced dot-matrix printers do a great job of reproducing these characters. Laser printers usually offer a special font family that includes the full ASCII set of characters. You'll have to experiment with your printer to determine whether or not it can output them.

Advanced DOS Commands

This section presents information on DOS commands that are not commonly used. Nevertheless, these commands have specialized, important functions that can be valuable additions to your DOS arsenal.

We have chosen four commands to discuss in this section. They are MODE, PROMPT, SET, XCOPY

MODE

Type: External

Syntax:

MODE n

MODE LPT#[:][n][,][m][,][P]

MODE [n],m[,T]

MODE COMn[:]baud[,][parity][,][databits][,][stopbits][,][P]

MODE LPT#[:]=COMn

MODE (device) CODEPAGE PREPARE=((codepage)
[d:][path]filename)

MODE (device) CODEPAGE PREPARE=((codepage list)
[d:][path]filename)

MODE (device) CODEPAGE SELECT=(codepage)

MODE (device) CODEPAGE [/STATUS]

MODE (device) CODEPAGE REFRESH

MODE sets the mode of operation for devices or communications. It can be used to set the mode for printers, monitors, or for the serial interface. It can be used to prepare and select code pages and to redirect printer output to the serial interface.

NOTE:
Some application programs that have to do with printers or communications (or special screen graphics) reset the MODE command as part of their normal operation. If such programs use a "startup disk," the program may reset the MODE parameters in an AUTOEXEC.BAT file or in another startup program. Sometimes these programs modify an existing AUTOEXEC.BAT file. These programs should automatically reset the correct screen or port attributes when they terminate; if they don't , you may have to reset the original attributes using the MODE command.

The MODE command can be used for several different types of operations. Each is summarized below.

Option 1. Printer Settings

Syntax:

MODE LPT#[:][n][,][m][,][P]

Discussion:

This option sets the output to the printer as either 80 or 132 characters per line. Unless this option is specifically set to 132, DOS uses the 80 characters per line setting for all printouts.

When any one of the DOS commands is used to send lines to the printer, this setting determines the MAXIMUM number of characters that will be sent before a line feed is sent to the printer to force the printing to continue on the next line.

Generally, the 132 characters-per-line setting is used only for wide-carriage printers. However, some printers can be reset to print very small characters. Therefore you can print 132 of these small characters across regular-sized printer paper (typically 8 1/2 by 11 inches). For more information about special printer controls, refer to the section, "DOS Printing Techniques" in this chapter.

Sets the printer number. You can enter 1, 2, or 3 to designate one of three printers.

n Sets the number of characters per line (80 or 132)

m Number of lines per inch (6 or 8) This option sets the output to the printer as either 6 or 8 lines per inch. Unless this option is specifically set to 8, DOS uses the 6 lines per inch setting for all printouts. The 8-lines setting results in noticeably less white space between printed lines (more lines of print squeezed into the same area).

> **NOTE:**
> When using the MODE command to set either the number of characters per line or the number of lines per inch, DOS will not warn you if you enter an illegal number. The command can only be used to set either 80 or 132 characters per line and only 6 or 8 lines per inch. If you enter any other numbers, there is no indication that the number has been refused; however, the current settings will not be changed.

Both the characters-per-line setting and the lines per inch are in effect when you use the DOS PRINT command to send output to the printer. However, if you use another printer program (for example, a printer program that is built into your word processor), these settings will probably be overridden. You can usually set both of these printing options by using your program's configuration options.

P Specifies the continuous retry option for time-out errors. This option places a portion of the MODE program in memory. It tells DOS to continuously try to send output to the printer if a time-out error occurs. This gives you a way to correct a printer error without terminating the printout. An example is when the printer returns an out-of-paper error. If you have used the P option, the computer will continue to try to send while you refill the paper supply. The printing will continue when the out-of-paper condition is resolved. If an error occurs that can't be so easily resolved, you can stop the continuous re-trying by pressing and holding the **Ctrl** key and pressing the **Break** key.

To change this condition, you must re-enter the MODE command with the printer options, but without using this P option.

Example:

To set printer number 1 to 132 characters per line and 8 lines per inch.

```
MODE LPT1,132,8
```

Option 2. Monitor Settings

Syntax:

MODE n

or

MODE [n],m[,T]

n should be one of the following expressions:

40	40 characters per line
80	80 characters per line
BW40	40 characters per line, black and white (color display adapter)
BW80	80 characters per line, black and white (color display adapter)
CO40	40 characters per line, color (color display adapter)
CO80	80 characters per line, color (color display adapter)
MONO	monochrome display adapter

Most computers include a switch on the system board that specifies the type of monitor you will be using. Other computers set the monitor display options using a special software program. Either way, the MODE command can usually be used to override this setting. If you change monitors, you can quickly reset the output format used by your display adapter by using this command.

You can shift the display left or right one character (40-character mode) or two characters (80-character mode) by specifying **R** or **L**.

When you use the T option, DOS will display a prompt that asks if the screen is aligned properly. If you enter **Y**, the command ends. If you enter **N**, the shift is repeated, followed by the same prompt.

Examples:

If you enter

mode 80,r,t

The display mode is set to 80 characters per line and the display is shifted two character positions to the right. The test pattern is displayed so that you can shift the display again without re-entering the command. It should look like this:

```
0123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789

Do you see the leftmost 0? (y/n)
```

Enter **y** if the first number on the line is 0; enter **n** if it is any other number. Each time you enter **n**, the display will be shifted two characters to the right.

If the characters need to be shifted to the left, enter

```
mode 80,1,t
```

The same alignment pattern will be displayed, but this time if you respond to the prompt with **n**, the display is shifted to the left.

If your monitor display needs aligning, you may want to put a form of this command in your AUTOEXEC.BAT file. If your display always needs to be shifted to the right, enter the following line into your AUTOEXEC.BAT file.

```
c:\dos\mode ,r
```

Each time this command is issued, the display will be shifted two characters to the right. Remember, MODE is an external DOS command. To use it in your AUTOEXEC.BAT file you must tell DOS where to find the MODE.COM program (in this case, in the DOS directory).

To increase the amount of the shift, repeat the command. For example, if your display must be shifted four characters to the right, enter the following two lines in your AUTOEXEC.BAT file:

```
c:\dos\mode ,r
c:\dos\mode ,r
```

Option 3. Asynchronous Communications Settings

Syntax:

```
MODE  COMn[:]baud[,][parity][,][databits][,][stopbits][,][P]
```

n 1 to 4 to indicate the asynchronous communications controller number (the communications port you want to use for communications). This will generally be set at 1 (unless your computer uses more than one output port).

The parity, databits, and stopbits settings depend on the control parameters required by your communications software.

baud 11 (110 baud), 15 (150 baud), 30 (300 baud), 60 (600 baud), 12 (1200 baud), 24 (2400 baud), 48 (4800 baud), or 96 (9600 baud)

parity N (none), O (odd), or E (even) (default = E)

databits 7 or 8 (default = 7)

stopbits 1 or 2 (for 110 baud, default is 2; for all other baud rates, default is 1)

P Specifies continuous retry option for time-out errors

These parameters are used to initialize the asynchronous communications controller. The baud rate is required, but for the other parameters, you may enter a comma to accept the default value.

As discussed above concerning the MODE option for printers, the **P** option tells DOS to continuously try to send output to the printer if a time-out error occurs. This gives you a way to correct a printer error without terminating the printout. You can stop the continuous re-trying by pressing and holding the **Ctrl** key and then pressing the **Break** key.

The **P** option is generally used when the asynchronous controller is being used with a serial interface printer.

Different hardware manufacturers may provide slightly different versions of this command.

Examples:

To set the mode of operation to 1200 baud, no parity, eight databits, and one stop-bit, enter

```
mode  com1:12,n,8,1,p
```

To accept the default values (even parity, 7 databits, and 1 stopbit) and use the "P" option, enter

```
MODE  COM1:12,,,,P
```

Option 4. Direct Parallel Printer Output (to an asynchronous communications controller)

Syntax:

```
MODE  LPT#[:]=COM n
```

This command redirects output directed to printer LPT# to asynchronous controller n.

You have to initialize the asynchronous controller with option 3 when you use option 4. You also have to specify the P parameter when a serial interface printer is connected to the RS-232C controller.

The printer number (1 to 4)

n The asynchronous communications controller number (1 to 4)

Option 5. To set MODE for Code Pages

This option is new to DOS versions 3.3 and later.. It offers a number of new functions related to the use of devices for country-specific operations. Some of the files on the DOS STARTUP disk for these versions are related to country-specific tasks. These files include some that have a ".CPI" filename extension; it is these filenames that are specified when using the country-specific options of the MODE command described below.

Syntax:

MODE (device) CODEPAGE

Used to display the current code page.

MODE (device) CODEPAGE REFRESH

Used to refresh the active current code page when the resident code page settings for a device have been lost. This can occur, for example, when a code page has been set for a printer and the printer loses power. The words "CODEPAGE REFRESH" can be abbreviated as "CP REFRESH."

MODE (device) CODEPAGE PREPARE=((codepage list)
[d:][path]filename)

or

MODE (device) CODEPAGE PREPARE=((codepage number)
[d:][path]filename)

Used to specify the prepared code page for the named device. The words "CODEPAGE PREPARE" can be abbreviated as "CP PREP."

MODE (device) CODEPAGE SELECT=(codepage number)
[d:][path]filename

This option is used to select the code page (which contains country-specific information) to be activated for the specified device. The words "CODEPAGE SELECT" can be abbreviated as "CP SELECT."

device	Used to specify one of these device identifiers: CON, PRN, LPT1, LPT2, LPT3.
codepage number	Used to specify the code page number (refer to Appendix B, "Country-Specific Information" for information on these code page numbers.

codepage list Used to specify the list of code page numbers (refer to Appendix B, "Country-Specific Information" for information on these code page numbers.

Examples:

To display the currently active code page for the device LPT1, enter

```
mode  lpt1 cp codepage
```

To re-establish a previously set code page for the device LPT1, enter

```
mode  lpt1 cp refresh
```

To specify three different code pages related to the CONSOLE device using the file "EGA.CPI," enter

```
mode  con cp prep=((860,,865) EGA)
```

Notice that the comma (without entering a code page number) is used to specify that the second code page previously prepared does not change.

To select the code page 850 for the printer LPT1, enter

```
mode  lpt1 cp select=850
```

The specified code page must be one of the code pages previously specified in the CODEPAGE PREPARE.

PROMPT

Syntax:

```
PROMPT  [prompt text]
```

This command provides a way to change the DOS system prompt. Normally, when DOS is waiting for input, it displays only the drive designation letter and a right angle bracket (for example, A>). By using the PROMPT command, you can change the prompt to provide additional information.

To change the DOS prompt, enter the name "PROMPT" followed by a space followed by one or more of the PROMPT options. Don't use spaces between the options.

The PROMPT character options are listed below (along with the prompt display they result in):

Option	Prompt
$b	The "|" character
$d	The current date
$e	The ESCAPE code
$g	The ">" character
$h	A backspace (erase preceding character, if any)
$l	The "<" character
$n	The current drive
$p	The current drive and directory or subdirectory
$q	The "=" character
$s	A leading space
$t	The current time
$v	The DOS version number
$_	A Carriage Return and Line Feed

Notice that in order to use any one of these options, you place a dollar sign ($) immediately in front of the identifying letter. If you don't enter the dollar sign, it tells DOS that any characters you enter should be displayed as part of the prompt.

One very useful DOS prompt displays the current drive designation letter followed by the name of the current subdirectory followed by the right angle bracket (>). You can invoke this prompt by entering

```
prompt $p$g
```

As a result, if A is the current drive and \DOS is the current directory, the DOS prompt will look like this

```
A:\DOS>
```

If you change to the "LETTERS" directory that is within the "SALES" directory that is within the root directory of drive C, the prompt will change to

```
A:\LETTERS\SALES>
```

You can use the PROMPT command to set up an almost infinite variety of command prompts and other on-screen effects.

For example, you can set up the prompt to send messages. If you enter

```
prompt  $p$g Station3$g
```

The resulting prompt displays the drive and directory, the right angle bracket, a space, the identifying message "Station3," and a closing right angle bracket. It would look something like this:

```
DOS> Station3>
```

You can also create a prompt that uses more than one line on the screen by using the "$_" code. For example, to create a prompt that displays the time on one line and then the usual drive and directory prompt on the following line, enter

```
prompt  $t$_$p$g
```

If you are working in drive A in the "\SALES" directory, the resulting prompt will look like this:

```
2:44:02.17
A:\SALES>
```

If the time option ($t) is included in the DOS prompt, each time it is freshly displayed, the time will be updated.

You can press the **Enter** key any time you want an update of the time. A series of presses of the **Enter** key would look like this:

```
1:24:04.52
1:24:05.29
1:24:05.84
1:24:06.50
```

You can use this capability to carry out simple timing sequences. For example, you could determine the amount of time it takes you to sort a list of addresses simply by noting the time displayed before and after you enter the SORT command. If you place the program name in a batch file, the time will be displayed when the program starts and when it ends.

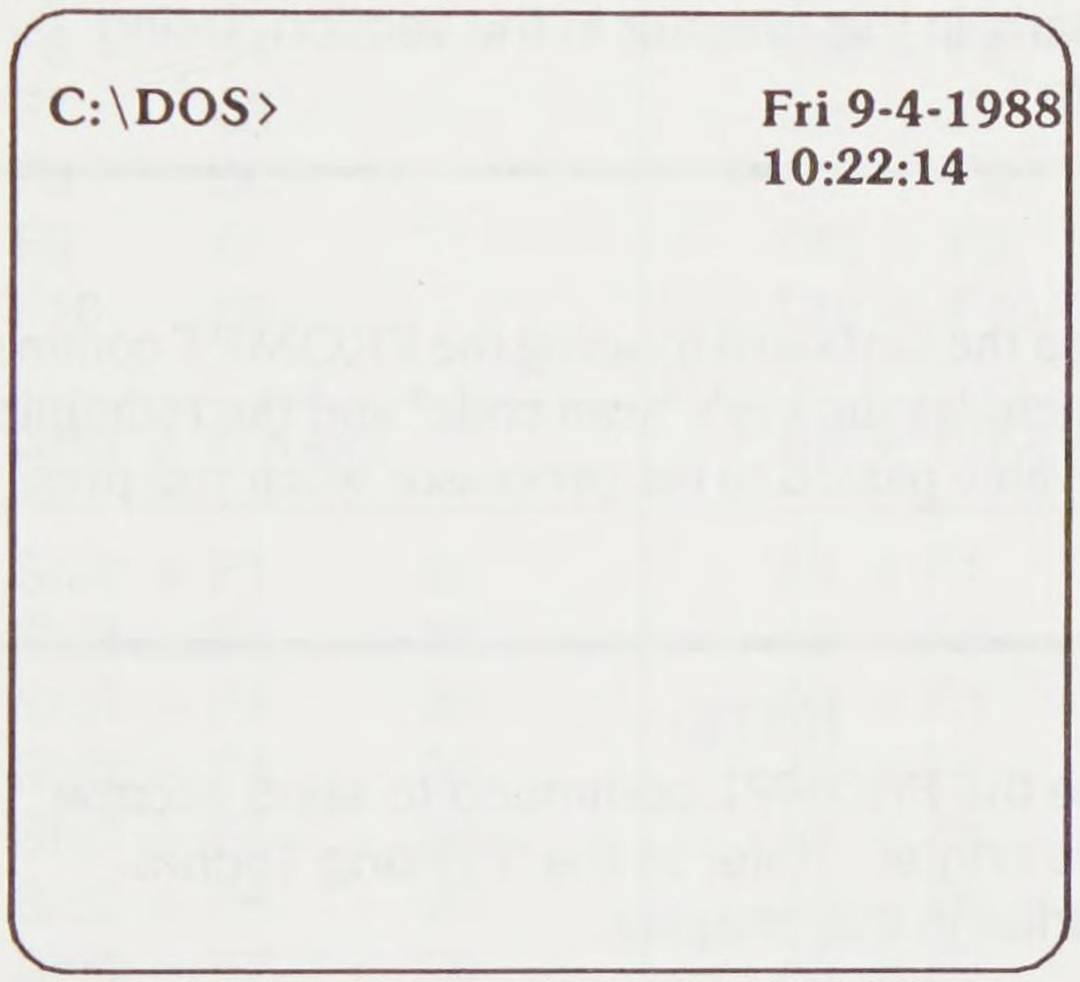
A Powerful PROMPT Sequence

A popular PROMPT sequence that places the date and time in the upper right corner of the screen is represented below.

prompt $e[s$e[1;67Hde[2;67Hthhh$e[u$p$g

You can place this prompt sequence in your AUTOEXEC.BAT file to load every time you start your computer (ANSI.SYS must specified in the CONFIG.SYS file).

This form of the PROMPT command uses a series of escape sequences ($e[) and backspace characters ($h) to place the date ($d) and time ($t) in the corner of the screen. Finally, the "pg" sets the DOS prompt to display the current drive and directory. If you are working in drive C in the "\DOS" directory when you use the above form of the PROMPT command, the resulting screen will look like this:



Using PROMPT to Program Keys

This section shows you how to create your own "macro-creating" methods using the PROMPT command. A number of programs are sold to create keystroke macros. These programs let you enter keystrokes and control functions and then assign this sequence to a key (or a key combination). But you can do the same thing using the PROMPT command.

One of the most powerful of the PROMPT command's capabilities is its ability to pass escape sequences. In the "Printer Control Using PROMPT" section earlier in this chapter, we discussed how to use the PROMPT command to pass the escape code and a printing style command to the printer. We can also use the PROMPT command to send escape sequences for both screen and keyboard control.

NOTE:

To use the PROMPT command to re-program keys, the keyboard/screen device driver (ANSI.SYS) must be loaded (for more information, refer to the Configuring Your System" section in this chapter). Make sure the following line exists in your CONFIG.SYS file:

DEVICE = ANSI.SYS

You can create the CONFIG.SYS file with this line in it using your word processor or using the COPY CON option described early in this chapter in the section "Using ASCII Characters."

You can redefine any key on the keyboard by using the PROMPT command to pass an escape sequence that includes the key's "scan code" and the redefinition for the key. The scan code is the value passed to the processor when you press a key.

NOTE:

You can also use the PROMPT command to send escape sequences to the printer. Refer to the "Printing Techniques" section earlier in this chapter.

By using the PROMPT command's capability to send an escape sequence, you can create keystroke macros for any key (and for some key combinations).

For example, you can redefine the computer's response when you press one of the "function keys" (the keys numbered **F1, F2, F3**, etc. at the left or at the top of your keyboard).

The scan code for function keys F1-F10 is listed in the table below. The scan code that will be passed when each function key is pressed in combination with the Shift, **Ctrl**, and **Alt** keys is also included since they are often used in macros.

F Keys		Ctrl + F Keys	
F1	59	Ctrl + F1	94
F2	60	Ctrl + F2	95
F3	61	Ctrl + F3	96
F4	62	Ctrl + F4	97
F5	63	Ctrl + F5	98
F6	64	Ctrl + F6	99
F7	65	Ctrl + F7	100
F8	66	Ctrl + F8	101
F9	67	Ctrl + F9	102
F10	68	Ctrl + F10	103
Shft + F Keys		**Alt + F Keys**	
Shift + F1	84	Alt + F1	104
Shift + F2	85	Alt + F2	105
Shift + F3	86	Alt + F3	106
Shift + F4	87	Alt + F4	107
Shift + F5	88	Alt + F5	108
Shift + F6	89	Alt + F6	109
Shift + F7	90	Alt + F7	110
Shift + F8	91	Alt + F8	111
Shift + F9	92	Alt + F9	112
Shift + F10	93		

You can use any one of these scan codes to create a macro definition for the related key or key combination. Enter the PROMPT command in this format:

prompt $e[0;(scan code);(key definition)p

The first set of characters entered after the PROMPT command ($e[) is how we enter the escape character. After a semicolon (used as a divider between command options), enter the scan code, another semicolon and then the definition for the key. If the key is to carry out more than one function, separate them with semicolons.

For example, to create a macro that uses the **F10** key to display the files in the current directory one screen at a time, enter

```
prompt $e[0;68;"DIR/P";13p
```

The "13" entered after the final semicolon sends the value of the **Enter** key and the "p" must be included to terminate the string.

After you enter this command, each press of the **F10** key will produce the display of the current directory.

Although this example will redefine the **F10** key, it will also redefine the current DOS prompt. Therefore you need to add, as the last step, the command for the DOS prompt you want to use. For example, to use the prompt that displays the current drive and directory, add the "pg" sequence to the line above. The final line will look like this:

```
prompt $e[0;68;"DIR/P";13p$p$g
```

After the F10 key is reset, the "pg" sequence is sent which resets the DOS prompt.

The most convenient way to use these macros is to put them in a batch file. You can, create a set of these small batch files to invoke the macro for each key. For example, you can store the above example as the two-line batch file named "F10.BAT." It is reproduced below.

```
echo off
prompt $e[0;68;"DIR/P";13p$p$g
```

Then, to redefine the **F10** key, all you have to do is enter the batch file name, "F10."

You may want to combine a series of these macro-definition lines in one batch file to redefine several keys.

NOTE:
These example re-definitions of function keys are active only when DOS is active (from the command line). When you load any other program that redefines these function keys, the definitions defined by that program will be in effect as long as the program is active. When you end that program and DOS is again active (anytime the DOS prompt is displayed), your function keys may have to be redefined by again calling your defining batch file.

Using PROMPT for Screen Control

Another form of the PROMPT command can be used to send the escape sequences for screen control. As in the previous example, the device driver "ANSI.SYS" is used and must be specified in the CONFIG.SYS file as

DEVICE = ANSI.SYS

Using this device driver, DOS provides the capability to set the screen colors and other attributes. Each color is associated with a number for foreground and background as follows:

Color	Foreground	Background
Black	30	40
Red	31	41
Green	32	42
Yellow	33	43
Blue	34	44
Magenta	35	45
Cyan	36	46
White	37	47

The other possible video attributes are:

Attribute	Number
Reset all	0
High Intensity	1
Mono Underline	4
Blink	5
Inverse	7
No display	8

To reset an attribute, use the PROMPT command to send the escape sequence followed by the attribute number. To send more than one attribute, divide them with a semicolon. For example, to reset the video colors to white letters on a blue background, enter

```
prompt $e[37;44m
```

The escape sequence ($e[) is sent followed by 37 (foreground white) and 44 (background blue). The trailing "m" tells DOS that the command is related to the display.

To reset the display to normal (white on black), enter

```
prompt $e[0m
```

You can combine more than one attribute in the same command format by separating each option with semicolons. To reset the display to high intensity with yellow letters on a red background, enter

```
prompt $e[1;33;41m
```

You can create a batch file to quickly reset the screen colors. An example is reproduced below.

COLORS.BAT

```
if x==%1x goto MESSAGE
echo Setting colors to [ 3%1 ] and [ 4%2 ]
prompt $e[0m
prompt $e[3%1;4%2m
prompt $p$g
echo off
goto END
:MESSAGE
echo off
echo No option entered
echo ...
echo Enter "COLORS" then 2 numbers.  Enter 1st number for
foreground
echo color, then a space, then enter 2nd number for background color
echo ...
echo Colors options are:
echo ...
echo 0  Black
echo 1  Red
echo 2  Green
echo 3  Yellow
echo 4  Blue
echo 5 Magenta
echo 6  Cyan
echo 7 White
:END
```

This batch file first checks to be sure at least one option was entered on the command line after the batch filename. If not, processing is transferred to the "LABEL" message where the color options are displayed. (If you need a reminder of the color numbers, just enter "COLORS" with no following options.) If any options were entered, they are displayed. Next, "prompt $e[0m" resets the existing screen attributes and then "prompt $e[3%1;4%2m" is activated using the values passed from the command line. Since foreground color settings must always begin with 3 and background settings must begin with 4, our PROMPT command already includes the 3 and 4 and we use the parameters passed to enter the second numeral of the value.

After the color attributes are established, "prompt pg" resets the normal DOS prompt (it displays the drive and directory) and the program ends. The color settings will not be affected; they will stay in effect until they are reset to another color or the 0 option is used to reset all attributes.

SET

Type: Internal

Syntax:

SET

or

SET string = string

The SET command is not difficult to use, but its use is not widely understood. When you use the SET command, the characters you enter are placed in the "environment." But, what (you might ask) is the environment? The answer is that it is an area of memory used by DOS to store data related to ongoing activity. It stores information related to DOS commands that use memory while they operate. The MODE and PRINT commands are examples. The environment also stores the location (in memory) of the command processor. Data needed by other DOS command are also stored in the environment: for example, if you have set a search path using the PATH command, this information must be saved in the environment.

If you enter

set

with no following characters, DOS will display the current contents of the environment that have to do with commands such as PATH and PROMPT. The name and location of the current command processor will also be displayed. A typical response might look something like the following:

```
COMSPEC=C:\COMMAND.COM
PATH=C:\DOS
PROMPT=$p$g
```

DOS also makes the environment space available for your use. You can place any string of characters in the environment by using the format

SET string1=string2

where both strings are sets of characters you enter. For example if you enter

set abc = xyz

That sequence of characters will be saved in the environment as

```
ABC = xyz
```

Notice that the first string you enter is converted to uppercase. Later, if you enter only the first string and the equal sign, both strings are removed from the environment. For example,

```
set  abc =
```

removes the "ABC = xyz" string from the environment.

The strings you save in the environment are up to you (as are the reasons for saving those particular strings). Why would you want to insert such strings in memory? The answer is: for the same reason DOS places strings there. That is, they are saved for later use. DOS needs the environment as a temporary storage area when, for example, it needs to know the search path previously set with the PATH command. You can use the strings you save in the environment for the same kinds of purposes. You can write your own programs that use these strings. You can also save strings that can be used later by application programs.

Probably the most useful way you can use these saved strings is in batch files. You can save strings and then use them later in much the same way you use a parameter passed to a batch program from the command line. In a batch file, you indicate that you want to use a string from the environment by placing percent signs immediately before AND after the string. When you specify the first part of the saved string (string1) in a batch file, it is replaced by the second part (string2) when the batch file runs. For example, if you save two path specifications by using the SET command as

```
set path1 = c:\letter\sales
set path2 = c:\files
```

you can use the strings later in a batch file. For example, the lines

```
echo off
copy a:file1.txt %path1%
```

in a batch file will copy "FILE1.TXT" to "C:\LETTER\SALES" and "FILE2.TXT" to "C:\FILES."

You can even use one batch file to save strings in the environment and use them later in another batch file. For example, the following batch file saves a string you pass from the command line for later use and again saves the path designation.

```
echo off
set tempfile = %1
set path1 = c:\letter\sales
```

The second line saves in the environment the parameter you pass from the command line as the name for "TEMPFILE." Later, a batch file that included this line

```
if not exist %1.tmp copy %1 %path1%%tempfile%
```

would copy a filename that matches the passed parameter to a temporary file (if the filename does not already exist with a .TMP extension).

XCOPY

Type: External

Syntax:

```
XCOPY [d1:] [PATH1] filename1 [d2:] [PATH2] [/A] [/D:(date)]
[/S [/E]] [/M] [/P] [/V] [/W]
```

The XCOPY command is a newcomer to the DOS set of commands. It was added as a better way to copy files (or groups of files) to and from directories using special copying options. This command is very useful for backing up data from a fixed disk.

As you can see, the command format is somewhat more complicated than the format for the COPY command. Each of the XCOPY options is described below.

d1: The source drive specification.

PATH1 The source path specification. If you omit this, the current subdirectory is assumed.

filename1 The source filename. This is not an option: you must enter at least the source filename.

d2: The destination drive specification.

PATH2 The destination path specification. If this is omitted, the current subdirectory is assumed.

/A Copy only files that have been set as archive files (using the ATTRIB command). The copied files will still be marked as archive files in BOTH the source and destination files.

/D:(date) Copy only those files in the source directory that have been changed on or later than the specified date.

/S Copy all files in the current directory or subdirectory and in any subdirectory under it.

/E Copy subdirectories, even if they are empty. (If you use this option, you must use the /S option also.

/M This copies the same marked archive files as specified in the /A option; however, the files on the destination disk are no longer marked as archive files.

/P Does not copy the file until you respond to a prompt that asks you to confirm that you want the file copied. However, if you are using the /E option, you will not be prompted before an empty subdirectory is copied.

/V Each file copied is verified to be sure the file stored on the destination disk is identical to the original on the source disk.

/W Copies only after you respond to a prompt to begin copying (if this option is omitted, the copying begins as soon as you enter the XCOPY command and press the **Enter** key.

There are also some special rules that are in effect whenever you use XCOPY:

- If the source is a directory, the directory will be copied onto the destination diskette.
- If the source to be copied includes more than one file, the destination files will be stored in a directory.
- You can add a backslash (\) to the end of the destination filename to make it a directory.

> **NOTE**
>
> If DOS does not find a directory on the destination disk that matches a PATH specification in the command format, the copying will stop and a message will be displayed asking you to specify whether or not the destination file specification is to be stored as a filename or a directory name.

As you can see, XCOPY is a much more powerful copying command than the COPY command. It is especially useful when backing up your fixed disk. A common usage would be to use the "/S" option along with the "/D:date" option to backup only the files that have been changed since the date of your last backup.

If you are using XCOPY for this purpose, you will probably want to make a copy of the program (XCOPY.EXE) in the root directory of your fixed disk (remember, when you use the /S option, XCOPY only copies files from subdirectories found UNDER the current directory.)

Chapter 6
DOS Command Reference

This chapter provides a reference to the MS-DOS set of operating commands. Each of the commands is listed in alphabetical order and each command section includes:

- Type - Indicates the type of the command (Internal or External).
- Syntax - Presents the form in which the command must be entered and the additional options, if any.
- Purpose - Briefly describes the function and uses of the command.
- Discussion - Describes completely the command's function and provides examples of how to use it.

DOS Command Syntax Notation

Each DOS command must be entered in a particular way: this command entry structure is known as the command's "syntax." In some user's guides, this may be referred to as a command format; however, in this book, "format" refers to the "syntax notation" (the way the command appears in print). For example, you can determine the items that are optional, by looking for information that appears inside square brackets. The notation [d:], for example, indicates an optional drive designation. The command syntax, on the other hand, is how YOU enter the command to make it work.

In Chapter 1, I described some of the basics of command entry, the things command syntaxes have in common. They are summarized in the next section.

Command Syntax Elements:

1. The DOS command name is the name you enter to start the DOS program (a few of the DOS commands can be entered using shortcut names). The DOS command name is always entered first. In this book, the command is usually printed in uppercase letters, but you can enter command names as either lowercase or uppercase or a mix of both.

2. Always leave a space after the command name.

3. The drive designation (abbreviated in this book as "d:") is an option for many DOS commands. However, some commands are not related to disk drives and therefore do not require a drive designation. Whenever you enter a DOS command that deals with disk drives and you are already working in the drive in question, you do not have to enter the drive designator.

   For example, if you are working in drive A (when the DOS prompt A> is showing at the left side of the screen) and you want to use the DIR command to display a directory listing of that same drive, you do not have to enter the drive designation. If you do not enter a drive designation, DOS always assumes you are referring to the drive you are currently working in (sometimes called the "default" drive).

> **NOTE:**
> You can use any of the DOS internal commands at any time, when working in any directory.

4. When referring to a drive in a DOS command, you must always follow the drive designator with a colon (:) (this is how DOS recognizes it as a drive designation).

5. A pathname (path) refers to the path you want DOS to follow in order to act on the DOS command. As described in Chapter 3, it indicates the path from the current directory or subdirectory to the files that are to be acted upon.

6. A filename is the name of a file stored on disk. As described in Chapter 1, a filename can be of eight or fewer letters or other legal characters.

7. A filename extension can follow the filename to further identify it. The extension follows a period and can be of three or fewer characters. A filename extension is not required.

8. Characters shown in a command syntax that are represented by a letter or number and preceded by a forward slash (for example, "/P") are command options (sometimes known as "switches"). Use of these options activate special operations as part of a DOS command's functions.

9. Items enclosed in square brackets are optional; in other words, the command will work in its basic form without entering the information contained inside the brackets.

10. Ellipses (...) indicate that an item in a command syntax can be repeated as many times as needed.

11. When items are separated by a vertical bar (¦), it means that you enter one of the separated items. For example:

    ON ¦ OFF

    This means that you can enter either ON or OFF, but not both.

Command Usage

When using a DOS command, type in the exact name and spacing indicated by the command format (we have printed a wide space to indicate where you leave a space).

Some commands are very simple and easy to use. For example, the SHIFT command has only one function (to shift the passed parameters in a batch file) and, therefore, there is only one form to enter. The command syntax is simply

SHIFT

This indicates that all you have to do is enter the command name (uppercase is not required) and then press the **Enter** key.

However, the CHDIR command (used to change the working directory) provides more than one command name that can be used to start the DOS program. The command syntax is

CHDIR (CD) [d:]path

This indicates that you can enter either CHDIR or CD to start the DOS program. Notice also that the format shows that one aspect of this command ([d:]) is optional. You must enter either CHDIR or CD to start the DOS program, then a space (notice the wide space printed in the format), and then the path for the new working directory; however, you not have to enter a drive designation letter. Whenever the drive designator is optional and you do not enter a drive, the command function will be carried out using the drive you are currently using.

Other commands provide more than one command capability. Each of the syntax descriptions for such commands are printed on a separate line. For example, the command syntax for the DATE command is printed as

DATE
DATE mm-dd-yy

This indicates that entering "DATE" alone results in a different function than entering the name followed by the numbers for a date (in the designated format). In this case, entering "DATE" will result in a display of the current date, followed by a prompt that tells you how to enter the date. The second form of the command provides a way to enter the date at the same time you enter the command name.

Same commands provide only options (no required entries). For example, the DIR command syntax is printed as

DIR [d:][path][filename][/P][/W]

Notice that every item listed after the command name is printed in square brackets. This indicates that they are all optional; in other words, the command will work with no options entered after the command name. If you do not enter a path or filename designation, it will carry out the command function (DIR displays the names of files and directories stored on disk) using the drive and directory you are CURRENTLY working in.

DOS Command Summary

The following summary briefly describes the syntax and purpose of the DOS commands in version 3.3. Included is the name of the command (including its short form, if any), the type of command (internal = I or external = E), the syntax of the command, and the function of the command.

APPEND E
APPEND [/X][/E]
APPEND [d:]path[;][d:]path[...]
APPEND ;

Displays or sets the search path for data files. DOS will search the specified path(s) if the file is not found in the current path).

ASSIGN E
ASSIGN x=y [...]

Redirects disk drive requests to a different drive.

ATTRIB [+R ¦ -R] [+A ¦ -A] [d:][path]filename [/S] E

Sets or displays the read-only and archive attributes of a file.

BACKUP d:[path][filename]d:[/S][/M][/A][/F][/P][/D:date] E
[/T:time][/L:[path]filename]

Makes a backup copy of one or more files.

BREAK I
BREAK ON ¦ OFF

Sets or displays the check for a **Ctrl** + **Break** key combination.

CALL [d:][path]batchfilename[options] I

Calls another batch file and then returns to current batch file to continue.

CHCP I
CHCP (codepage)

Displays the current code page or changes the code page that DOS will use.

CHDIR (CD) [d:]path [I]

Displays working (current) directory and/or changes to a different directory.

CHKDSK [d:][path][filename][/F][/V] [E]

Checks a disk and provides a file and memory status report.

CLS [I]

Clears (erases) the screen.

COMMAND [E]
COMMAND [d:][path][device][/P][/C (string)][/E:(size)]

Starts a new version of the MS-DOS command processor (the program that loads the DOS internal programs).

COMP [d:][path][filename] [d:][path][filename] [E]

Compares two groups of files to find information that does not match.

COPY [d:][path]filename [d:][path][filename][/A][/B][/V] [I]
COPY [/A][/B][/V] [d:][path]filename
or (to append)
COPY [d:][path]filename+[d:][path]filename[...]

Copies and appends files.

CTTY (device) [I]

Changes the standard I/O (Input/Output) device to an auxiliary device.

DATE [I]
DATE mm-dd-yy

Displays and/or sets the system date.

DEL (ERASE) [d:][path]filename [I]

Deletes (erases) files from disk.

DIR [d:][path][filename][/P][/W] [I]

Displays directory of files and directories stored on disk.

DISKCOMP [d:] [d:][/1][/8] [E]

Compares the contents of two diskettes.

DISKCOPY [d:] [d:][/1] E

Makes an exact copy of a diskette.

ECHO ON ¦ OFF
ECHO (message) I

Displays messages or turns on or off the display of commands in a batch file.

EXE2BIN [d:][path]filename [d:][path]filename E

Converts .EXE (executable) files to binary format.

EXIT I

Exits a secondary command processor.

FASTOPEN d:[=n] E

Keeps track of the locations of files for fast access.

FC [/A][/C][/L][/Lb n][/N][/T][/W][number] [d:][path]filename [d:][path]filename E

or (for binary comparisons)

FC [/B][/number] [d:][path]filename [d:][path]filename

Displays the differences between two files or sets of files.

FDISK E

Prepares a fixed disk to accept MS-DOS files for storage.

FIND [/V][/C][/n] "string" [d:][path]filename[...] E

Finds and reports the location of a specific string of text characters in one or more files.

FOR %%(variable) IN (set) DO (command) I
or (for interactive processing)

FOR %(variable) IN (set) DO (command)

Performs repeated execution of commands (for both batch processing and interactive processing).

FORMAT d:[/1][/2][/4][/8][/n:sectors][/n:tracks][/B][/S][/V] E

Formats a disk to accept DOS files

GOTO (label) [I]

Causes unconditional branch to the specified label.

GRAFTABL (code page) [E]
GRAFTABL /STATUS

Loads a table of character data into memory (for use with a color/graphics adapter).

GRAPHICS [printer type][/B][/R][/LCD][p=port] [E]

Provides a way to print contents of a graphics screen display.

IF [NOT] EXIST filename (command) [I]
IF [NOT] (string1)==(string2) (command)
IF [NOT] ERRORLEVEL (number) (command)

Allows for conditional operations in batch processing.

JOIN [E]
JOIN d: d: path
JOIN d:/D

Allows access to the directory structure and files of a drive through a directory on a different drive.

KEYB
KEYB [xx][,][yyy][,][d:][path]filename

Loads a program that replaces the support program for U. S. keyboards.

LABEL [d:][volume label] [E]

Creates or changes or deletes a volume label for a disk.

MKDIR (MD) [d:]path [I]

Creates a new subdirectory.

MODE n [E]
MODE LPT#[:][n][,][m][,][P]
MODE [n],m[,T]
MODE COMn[:]baud[,][parity][,][databits][,][stopbits][,][P]
MODE LPT#[:]=COMn
MODE (device) CODEPAGE PREPARE=((codepage) [d:][path]filename)
MODE (device) CODEPAGE PREPARE=((codepage list) [d:][path]filename)
MODE (device) CODEPAGE SELECT=(codepage)
MODE (device) CODEPAGE [/STATUS]
MODE (device) CODEPAGE REFRESH

Sets mode of operation for devices or communications.

MORE < (filename or command) [E]
(name) ¦ **MORE**

Sends output to console, one screen at a time.

NLSFUNC [d:][path]filename [E]

Used to load a file with country-specific information.

PATH [I]
PATH [d:]path[;][d:]path[...]

Sets or displays directories that will be searched for programs not in the current directory.

PAUSE [comment] [I]

Suspends execution of a batch file until a key is pressed.

PRINT [E]
PRINT [/B:buffersize][/D:(device)][/M:(maxtick)][/Q:(value)][/S:(timeslice)] [/U:(busytick)][/C][/P][/T] [d:][path][filename] [...]

Queues and prints data files.

PROMPT [prompt text] [options] [I]

Changes the DOS command prompt.

RECOVER [d:][path]filename [E]
RECOVER d:

Resolves sector problems on a file or a disk.

REM [comment] [I]

Provides a way to insert remarks (that will not be acted on) into a batch file.

RENAME (REN) [d:][path]filename [d:][path]filename [I]

Changes the filename under which a file is stored.

REPLACE [d:][path]filename [d:][path][/A][/P][/R][/S][/W] [E]

Replaces stored files with files of the same name from a different storage location.

RESTORE d: [d:][path]filename[/P][/S][/B:mm-dd-yy][/A:mm-dd-yy] [/E:hh:mm:ss][/L:hh:mm:ss][/M][/N] [E]

Restores to standard disk storage format files previously stored using the BACKUP command.

RMDIR (RD) [d:]path [I]

Removes a subdirectory.

SELECT [d:] [d:][path][country code][keyboard code] [E]

Formats a disk and installs country-specific information and keyboard codes.

SET [I]
SET (string1)=(string2)

Inserts strings into the command environment. The set values can be used later by programs.

SHARE [/F:space][/L:locks] [E]

Installs support for file sharing and file locking.

SHIFT [I]

Increases number of replaceable parameters to more than the standard ten for use in batch files.

SORT [/R][/+n] < (name) [E]

Sorts input and sends it to the screen or to a file.

SUBST d: d:path [E]
SUBST d:/D

Substitutes a virtual drive letter for a path designation.

SYS d: [E]

Transfers the operating system files to another disk.

TIME I
TIME hh:mm[:ss][.cc]

Displays current time setting of system clock and provides a way for you to reset the time.

TREE [d:][/F] E

Displays directory paths and (optionally) files in each subdirectory.

TYPE [d:][path]filename I

Displays the contents of a file.

VER I

Displays the DOS version number.

VERIFY I
VERIFY ON ¦ OFF

Turns on the verify mode; the program checks all copying operations to assure that files are copied correctly.

VOL [d:] I

Displays a disk's volume label.

XCOPY [d:][path]filename [d]:[path][filename] E
[/A][/D:(date)][/E][/M][/P][/S][/V][/W]

Copies directories, subdirectories, and files.

APPEND

Type: External

Syntax:

APPEND

APPEND [/X][/E]

APPEND [d:]path[;][d:]path[...]

APPEND ;

Purpose: Sets the path to be searched for data files or displays the current search path.

Discussion

The APPEND command is new to DOS with versions 3.3 and later. It gives you a way to set the search path for data files.

The APPEND command is similar to the PATH command that tells DOS where to search for program files (files with a .COM, .EXE, or .BAT filename extension). The APPEND command guides the search for data files (such as text files).

> **NOTE:**
> If a file with a .COM, .EXE, or .BAT filename extension is to be treated as a data file (for example, if you want to edit the contents of a batch file), its path CAN be specified using the APPEND command. However, if you want to execute the file from any directory, you must specify its location using the PATH command.

Using APPEND followed only by a space and a semicolon (APPEND ;) deletes previously appended search paths. DOS will then search for data files only on the current drive and directory.

Use the APPEND command to tell DOS where to search for data files if a specified file is not found in the current directory. This means that you will need only one copy of a file on your fixed disk, even if you use it for different purposes. For example, you can store a copy of the file "NAMES1.TXT" in the directory "LISTS" and use it (copy from it, insert it into other files) while working in any drive or directory.

Use the APPEND command without options to display the current search path. Invalid drive designations will not be detected until DOS attempts to use the search path to find a specified file.

Once APPEND has been loaded into memory, you can re-enter the APPEND command to change the specifications from any drive or directory.

The APPEND command CAN be used on a network.

NOTE:
If you have used APPEND with the /X option and you plan to use the BACKUP and RESTORE commands, you must first cancel the option by entering

append ;

If you plan to use the ASSIGN command to redirect drive requests, use the APPEND command first (refer to the ASSIGN command section for more information).

Options

/X The /X switch is used by DOS to guide the search path for some of the other DOS commands (it is generally used only by programmers). Set the /X switch before specifying paths to search. The /X switch can only be specified the first time you use the APPEND command.

/E Use the command with the /E switch to store the names of the APPENDed directories in the DOS environment (refer to the SET command for more information about the DOS environment). The /E switch can only be specified the first time you use the APPEND command.

If you want DOS to automatically search for files stored on drive C in either the "MEMOS" subdirectory or the "LETTERS" subdirectory, enter

append c:\memos;c:\letters

DOS will first search for any data files you specify in the currently active drive and directory. If the files are not found there, it will search the path \MEMOS, and finally the path \LETTERS.

For example, if you want to use the TYPE command to display the contents of the file "LET3-9.TXT" that is in the "\LETTERS" directory, you can enter

type let3-9.txt

DOS will first search for the file in the current directory, but if it is not found there, DOS will immediately search the "\MEMOS" and "\LETTERS" directories on drive B. If the file is found there, the TYPE command will immediately display its contents.

If you want DOS to search additional directories, you must enter the entire command again, listing all the directories you want to search. Each path designation must be separated by a semicolon. For example, if you want DOS to continue to search the "\MEMOS" and "\LETTERS" directories, but want to first search for specified data files on drive B, re-enter the command like this:

append b:;c:\memos;c:\letters

ASSIGN

Type: External

Syntax:

ASSIGN

ASSIGN x=y [...]

Purpose: Redirects disk drive requests to a different drive.

Discussion

Use the ASSIGN command to change drive assignments from the drive indicated by the first letter you enter (x), to the drive indicated by the second letter (y) entered. Do not enter a colon after the drive letters.

Entering the command without any parameters causes all drive reassignments to be reset to normal. The equal sign is optional.

This command is especially useful when you are using a program that automatically attempts to read or write to a specified drive. By reassigning that drive letter, you tell DOS to ignore requests to the first drive letter and instead carry out both reads and writes on the second drive letter. When you want DOS to again carry out drive requests as usual, enter "ASSIGN" without any options.

You can use this command with application programs that attempt to act on drives you don't want to use (or don't have installed in your computer).

This command should not be used with other DOS commands that need to get drive information. The FORMAT, DISKCOPY and DISKCOMP will ignore any drive reassignments.

Also, don't use the BACKUP, RESTORE, LABEL, JOIN, SUBST, or PRINT COMMANDS with re-directed drives.

Example

```
assign  a=b
```

DOS sends any requests for drive A to drive B. After this form of the command is entered, the command

```
dir  a:
```

will result in a display of the files on drive B instead of drive A.

You can reassign more than one drive at a time (leave a space between entries). For example, enter

```
assign  a=b  c=b
```

Requests for either drive A or drive C would automatically be sent to drive B. For example, if you enter

```
dir  a:
```

DOS will display a directory of the files on drive B (not of drive A).

ATTRIB

Type: External

Syntax:

ATTRIB [+R | -R] [+A | -A] [d:][path]filename [/S]

Purpose: Sets or displays the read-only and archive attributes of a file.

Discussion

Using the ATTRIB command, you can change a file's read/write attribute or set the archive attribute. If you use this command to specify a file as "read-only," the file can be accessed, but not altered or deleted.

If a file has an attribute of -R, it can be both read from or written to (it is referred to as "read/write"). If a file has an attribute of +R, it can be read from, but not written to (for example, you can use your word processor to look at it but not to alter it).

NOTE:
If you try to erase a file after you have set its attribute to read-only (+R), DOS will display

Access denied

Before you can erase the file you must use the ATTRIB command to change the file's attribute to read/write (-R).

The BACKUP, RESTORE, and XCOPY commands can use the archive attribute. You can use the **+a** or **-a** options to selectively back up using the BACKUP command. You can also use these options to selectively copy using the XCOPY command.

Wildcard characters (? and *), can be used. However, you CANNOT use a directory name instead of a filename to change the attributes of all files in a directory.

Options

+R Use the +R option to make a file read-only. Read-only files may be read but they can't be changed or deleted.

-R Use the -R option to change the file protection attribute back to normal (so it can be read, changed, or deleted).

+A Use the +A option to set the ARCHIVE attribute of a file. When the +A option is used, this "flags" the file as available for archiving when using the BACKUP or XCOPY commands.

-A Use the -A option to turn off the archive attribute.

/S Use the /S switch to set attributes on subdirectories found within the specified path.

Examples

The following form of the command sets the read-only attribute for the file SALES87.TXT:

```
attrib +r a:sales87.txt
```

After you have set the read-only attribute for the file "SALES87.TXT," DOS will not allow you to delete or modify it (until the attribute is removed).

To set the archive attribute for the file "LETTER3.DOC" which is in the "\LETTERS" directory on drive B, enter

```
attrib +a b:\letters\letter3.doc
```

After the archive bit has been set, commands such as XCOPY can selectively act on them (using the BACKUP or XCOPY /M switch). For more information, refer to the BACKUP and XCOPY command sections.

To determine the current attribute settings for all of the files in the current directory, enter

```
attrib *.*
```

The attribute settings for all of the files on the current directory will be displayed. If, for example, you were working in the "\LETTERS" directory on the diskette in drive B, ATTRIB would display something like this:

```
A       B:\LETTERS\LETTER3.TXT
A  R    B:\LETTERS\SALES87.DOC
        B:\LETTERS\SALES-N.TXT
```

This display indicates that three files were found and that the first two had the archive bit set. The second file is also marked as read-only.

BACKUP

Type: External

Syntax:

BACKUP [d:][path][filename] [d:][/S][/M][/A][/F][/P][/D:date] [/T:time][/L:[d:][path]filename]

Purpose: Makes a backup copy of one or more files.

Discussion

The first path and filename(s) you enter identify the disk file(s) to be backed up. The second drive specified is the drive where the backup copy will be sent.

If you do not specify the first path, files are backed up from the current directory. If you do not specify a filename or extension, then all files in the directory are backed up.

You can use the wild card characters (* and ?) to specify groups of files to be backed up.

If errors are encountered during the BACKUP process, the ERRORLEVEL exit code is set as follows:

1 No files could be found to back up.
2 Some files were not backed up due to file conflicts.
3 Terminated when you pressed the **Ctrl** + **Break** key combination.
4 Terminated due to any other type of error.

These codes can be used with the batch processing IF command to write backup batch files that test for errors during the backup process.

Files backed up using the BACKUP command are stored in a special, compacted format and are therefore not usable for normal file processing. You must use the RESTORE command to recall them and store them in the normal (un-compacted) format.

NOTE:
DOS versions 3.3 and later store backed up files in a different format than earlier versions (the new backup method requires less disk space). In these versions, the BACKUP program creates two files on the backup disk: "BACKUP" and "CONTROL." The "BACKUP" file will contain all the files that were backed up and the "CONTROL" file will contain file storage information such as the paths and filenames of the source files.

Options

/A Does not erase the files on the target disk (DOS normally will erase existing files on the target diskette before it starts backing up the files). Use of this option cancels the prompt to insert a diskette in the target drive before the copying starts, but after the target disk is filled with backed-up files, you will be prompted to insert a new diskette.

NOTE:
The /A option does not work if the files on the backup disk were copied using a BACKUP program version of 3.2 or earlier.

/F Formats the target disk (if it is not already formatted). This option uses the FORMAT program which must, therefore, be accessible via the current path.

/L Makes an entry in the backup log in the file you specify here. If you use this option but do not specify a backup log file, the system creates the file BACKUP.LOG in the root directory of the source drive.
The backup log will contain:

1. The date and time that the files were backed up.
2. The name of the file that was backed up.
3. The number of the backup disk that contains the file.

The information stored in this file can be used when you want to restore a particular file from a backup disk.

If the backup log file that you specify already exists, the current entry is added to the existing file.

/M Only backs up files that have been modified since the last time the BACKUP command was used. This switch checks the archive attribute of a file. If the file's archive attribute is set to off (-A), the file will not be backed up.

> **NOTE:**
> If you are backing up files from a diskette, do not write-protect the it. BACKUP changes the archive attribute of each file as it is backed up.

/P Packs as many files as possible onto each diskette. This option will create a directory when that is the only way to pack more files onto the diskette.

/S Causes files in the specified directory and all subdirectories below the specified directory to be backed up.

/D:(date) Backs up files only if they have been created or modified on or after the date you enter.

/T:(time) Backs up files only if they have been created or modified on or after the time you enter.

Example

In the following example, all files in the drive C "LETTERS" directory that begin with "SALE" and with any filename extension are backed up onto the disk in drive A.

```
backup c:\letters\sale*.* a:
```

BACKUP will display a prompt when it is time to insert the disks. If the /A option is not used, there will also be a warning that any files currently stored on the target disk will be erased.

BREAK

Type: Internal

Syntax:

BREAK

BREAK ON | OFF

Purpose: Instructs DOS to check for a Ctrl + Break key combination.

Discussion

DOS normally detects a press of the **Ctrl + C** key combination only while checking for keyboard input or while sending characters to the screen.

Some application programs automatically reset BREAK (either ON or OFF); this will cancel the BREAK options you set.

Examples

Enter

break on

to tell DOS to check for the **Ctrl + C** key combination.

To determine the current BREAK setting, enter

break

CALL

Type: Internal

Syntax:

CALL [d:][path]batchfilename[options]

Purpose: Calls another batch file and then returns to the current batch file to continue processing.

Discussion

Used within a batch file to specify the name of another batch file (a file with the .BAT filename extension). The original batch file calls into action the CALLed batch file. When the CALLed batch file ends, control is transferred back to the original batch file.

This command function (transfer of control back to a calling batch file) was not available prior to MS-DOS versions 3.

Example

To run the files STARTER.BAT, TESTER.BAT, and FINISH.BAT in sequence, enter the following three lines in your batch file:

```
call starter
call tester
call finish
```

CHCP

Type: Internal

Syntax:

CHCP

CHCP (codepage)

Purpose: Displays or changes the code page that DOS will use.

Changes the code page used by DOS for device-related system operations such as displaying or printing special language characters. (The code page is a table that defines the character set for a country.)

Discussion

To use this command, the code page number must have been previously set (refer to Appendix B for more information).

After this command sets a new code page, DOS will access the "COUNTRY =" command in the CONFIG.SYS file (when the computer is started) to set up special foreign language characters for display or printing. Programs run after using this command to change the code page will use the new code page.

The NLSFUNC command must be used prior to using this command. Also, device drivers must be prepared in advance (refer to the MODE command).

If you select a code page that has not been prepared for your system, DOS will display an error message similar to the following:

```
Code page xxx not prepared for system
Active code page: xxx
Prepared system code pages: xxx  xxx
```

The Xs will be replaced by the appropriate number.

If you change code pages and a device has not been prepared for use with that code page, the code page will still be changed. However, DOS will display an error message similar to the following:

Code page xxx not prepared for device xxx.

The Xs will be replaced by the appropriate number.

More information on using code pages can be found in Appendix B.

Examples

If you enter "CHCP" without entering a number, the program will display a message similar to the following:

```
Active code page  437
Prepared code pages  437  863
```

To change the code page setting from 437 to 863, enter

```
chcp  863
```

CHDIR (CD)

Type: Internal

Syntax:

CHDIR (CD) [d:]path

Purpose: Displays working (current) directory and/or changes to a different directory.

Discussion

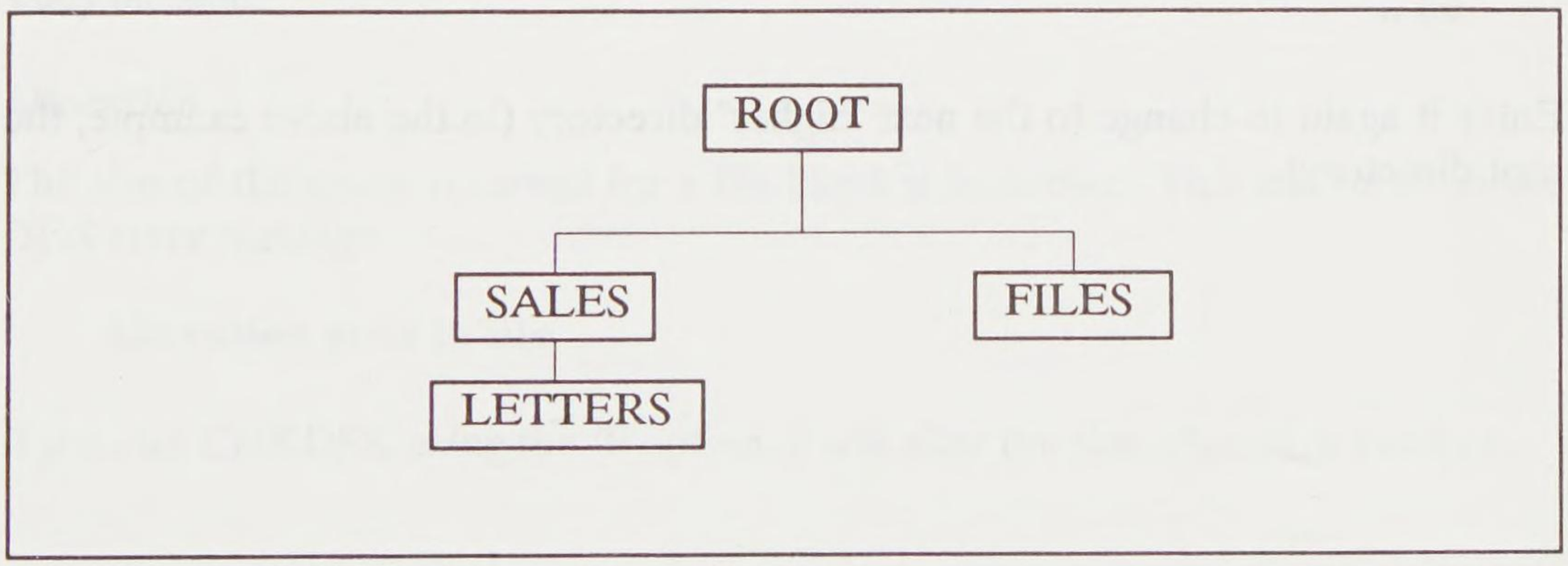
Used to change from one directory to another you specify. The command can be used without the drive designator (d:) if you were changing to a different directory on the current drive. The path designator can indicate a subdirectory within another directory or subdirectory, but it must not exceed 63 characters in length.

Enter "CHDIR" or "CD" without a drive and path designation to get a display of the current directory.

For more information on using CHDIR and other fixed-disk commands, refer to Chapter 3, "Using Fixed Disks."

Examples

To demonstrate how the CHDIR command accesses the DOS directory structure, let's look at an example directory structure that looks like this:



As indicated in this diagram, two directories, "SALES" and "FILES" are to be found "within" the root directory (indicated by the backslash). An additional directory, "LETTERS" is to be found within the SALES directory.

To change to the directory "LETTERS" (that is within the directory "SALES") on the root directory of drive C, enter

```
cd\sales\letters
```

The command indicates the path sequentially "down through" the directory path: the first backslash indicates the root directory, down through the directory "SALES," down to the target directory "LETTERS."

If you are already working in the SALES directory, you can change back to the root directory simply by entering

```
cd\
```

To change to a directory path other than through the root directory, do not enter the leading backslash. For example, if you are working in the "SALES" directory, you can change to the "LETTERS" directory without going through the root directory by entering

```
cd letters
```

A shortcut method to change to the directory immediately "above" the current directory allows you to enter the CHDIR command followed by two periods. For example, to change from the "LETTERS" directory to its "parent" directory, "SALES" enter

```
cd ..
```

Enter it again to change to the next "higher" directory (in the above example, the root directory).

CHKDSK

Type: External

Syntax:

CHKDSK [d:][path][filename][/F][/V]

Purpose: Checks a disk and provides a file and memory status report.

Discussion

Checks for errors on a disk. Displays error messages (if problems are found) and issues a status report.

If you specify the /F option, the program will correct non-contiguous errors when it finds them. These errors can be accidentally created due to the fact that DOS does not always store files in one continuous block on the disk. If space is limited on the disk (especially if you have previously erased and replaced files), DOS may store the file in two or more pieces in whatever spaces are available on the disk. If the links between the clusters (blocks of data) are lost, CHKDSK will discover the error and report it. By using the /F option, you tell CHKDSK to find any of these parts of files or programs that have become separated from the rest of the file, to create a new file for each, and to write the lost segment to these files. These new files will all start with the letters "FILE" and will end with four numbers (starting with 0000) plus a ".CHK" extension. The first file created will be labeled FILE0000.CHK.

When you use the /F option, CHKDSK also looks for and corrects other types of disk errors (refer to Appendix A for more information about CHKDSK errors). They include:

Allocation

The size of the space reserved for a file block is incorrect. This can result in the DOS error message

Allocation error in file

If you run CHKDSK using the /F option, it will alter the size allocation number.

Attribute

There are a number of possible errors having to do with DOS's internal record of a file's attributes. For example, a size of linking attribute associated with one or more files does not match the actual file information. This can result in the error message

Entry has a bad attribute

If you run CHKDSK using the /F option, the program will try to correct the error.

First Cluster Number Invalid

DOS's internal directory of file information on the disk contains a pointer to file storage information that does not match the actual files stored on the disk. This can result in the error message

First cluster number is invalid, entry truncated

If you run CHKDSK using the /F option, the pointer information is corrected (truncated to a zero length file).

The CHKDSK command was also discussed in Chapter 1, "Introduction."

Options

/F Corrects errors when it finds them. If CHKDSK finds lost clusters (parts of files or programs that have become separated from the rest of the file) it will write the lost segment to new files and provide the filename "FILEnnnn" (nnnn will be a number starting with 0000).

/V Displays progress messages while CHKDSK is in operation.

filename If you enter a filename, CHKDSK also reports how many files are stored in non-contiguous blocks (see explanation of non-contiguous blocks above).

Examples

If you enter

```
chkdsk a:
```

CHKDSK will analyze the disk in drive A and report a status message. It will look something like this:

```
362496    bytes total disk space
53248     bytes in 2 hidden files
301056    bytes in 8 user files
8192      bytes available on disk

524288    bytes total memory
198976    bytes free
```

The first four lines of this display refer to the disk you specified: they indicate the total disk space (the storage capacity of the disk), the number of hidden (system) files stored on the disk, the number of user files stored on the disk, and the amount of storage space still remaining to be used. If you had created directories on the disk, they would be reported in a fifth line. The last two lines do not refer to the disk you specified; they indicate the amount of memory (RAM) in your system and how much of it remains available for use.

If (for example) three lost clusters are found, the program will also display

```
3 lost clusters found in 3 chains.
Convert lost chains to files (Y/N)?
```

After this error message is displayed, you should run CHKDSK again, this time using the /F option. Enter the command as

```
chkdsk  a: /f
```

Three new files would be created on the root directory of the disk with the following filenames:

```
FILE0000.CHK
FILE0001.CHK
FILE0002.CHK
```

Each of these files will contain a lost cluster. You can use your word processor to look at these files to determine which file the lost segment came from. If the lost segment came from a text file you may be able to re-merge it with the main file. However, upon inspection, you may find that the original text file is intact. This may indicate that the lost segment resulted from file management tasks carried out by your word processor (or other type of program). In this case, you can simply delete the newly created segment file.

If the lost segment came from a program file, you probably will not be able to re-merge it. If possible, replace the file by re-copying from the original source. If you cannot replace the file, check the program file that is stored on disk carefully by running all of its features. It may still be operational.

CLS (Clear Screen)

Type: Internal

Syntax:

CLS

Purpose: Clears (erases) the screen.

Discussion

Erases all characters and graphics from the screen; however, it does not change the currently-set screen attributes.

Example

Enter

cls

to clear the screen.

COMMAND

Type: External

Syntax:

COMMAND

COMMAND [d:][path][device][/P][/C (string)][/E:(size)]

Purpose: Reloads the DOS command processor (the program that loads the DOS internal programs) or starts a new command processor.

Discussion

Using this command with no options loads a new command processor with the DOS internal programs. Starting a new command processor also creates a new command environment (SET commands are canceled).

The command processor is loaded in two parts: the transient component and the memory-resident component. The memory-resident part resides in a portion of memory that can occasionally be overwritten by application programs. If that happens, DOS will discover that the memory-resident portion is missing and it will attempt to go to the disk to reload the transient portion from the COMMAND.COM program. If COMMAND.COM cannot be found, DOS will terminate operations until the COMMAND program can reload.

Options

device Specifies a new input/output device at the time the new command processor is loaded.

/C (string) Performs the command indicated by the string and then returns to the original command processor.

/E:(size) Specifies the environment size. The size is indicated by n and can be from 160 to 32,768 bytes.

/P Keeps the secondary command processor in memory from that point in time forward. The only way to reload the original command processor is by restarting the system.

Example

If you enter

```
command /c dir
```

DOS will load the secondary command processor, execute the DIR command to show the files on the current drive and directory, and then exit back to the original command processor.

COMP

Type: External

Syntax:

COMP [d:][path][filename] [d:][path][filename]

Purpose: Compares two groups of files to find information that does not match.

Discussion

Compares two files or two groups of files. The files can have the same names, or different names. Wildcard characters (? and *) can be used to specify groups of files to be compared (files of the same names will be compared).

Each set of specified files will be compared, one set at a time. Before starting the comparison, the program checks to see if files contain a valid end-of-file mark. COMP will not compare files that do not contain an end-of-file marker. If a specified file does not contain such a mark, the following error message will be displayed:

EOF mark not found

This mark is created by your word processor or other application program to indicate the end of the file. You can create the end- of-file marker by pressing and holding the **Ctrl** key and then pressing the letter **Z** key.

The COMP program also checks to make sure the file sizes indicated in the directory are the same for compared files; a comparison is not carried out if the files are stored as different sizes.

If you enter one or more pathnames or filenames after the command name, when you press the **Enter** key the comparison will begin immediately; therefore, you should make sure the diskette(s) containing the files to be compared are already in the specified drives. If you want to change diskettes after entering the command, enter the command name with no following parameters. If you enter the COMP command with no options, the program will prompt you for the file and/or path designations. First, the program will display:

Enter primary filename

Enter the name of the first file or group of files. Next, the program will display:

Enter 2nd filename or drive ID

After you enter the second filename(s), the program will compare the files, issuing reports on its progress.

If the files are identical, it will report:

Files compare OK

If the files are not identical, it will report on the differences it finds.

COMP will display the memory locations of any portions of the compared files that do not match. After ten mismatches, the program terminates the comparison. The following message is displayed

10 Mismatches - ending compare

After all pairs specified have been compared, COMP will display

Compare more files (Y/N)?

To enter more files to compare, press the letter **Y** key. To terminate the comparisons, press the letter **N** key.

Examples

To compare all the files ending in ".TXT" on drive A with all of the files ending in ".TXT" on drive B, enter

comp a:*.txt b:

To compare all of the files in the "LETTERS" directory on drive A with all of the files in the "LETTERS" directory on drive B, enter

comp a:\letters b:\letters

COPY

Type: Internal

Syntax:

COPY [d:][path]filename [d:][path][filename][/A][/B][/V]

COPY [/A][/B][/V] [d:][path]filename

or (to append)

COPY [d:][path]filename+[d:][path]filename[...]

Purpose: Copies or appends files. Files can be copied with the same name or with a new name.

Discussion

COPY is usually used to copy one or more files from one loaction to another. However, COPY can also be used to create new files. By copying from the keyboard console (COPY CON:) to the screen, files can be created and then saved to disk.

The first filename you enter is referred to as the "source" file. The second filename you enter is referred to as the "target" file. If errors are encountered during the copying process, the COPY program will display error messages using these names.

Unlike the BACKUP command, copied files are stored in the same format as they are found in. The copied files can be used just as you would use the original (whether the copied file is a data file or a program).

COPY can also be used to transfer data between any of the system devices. Files may also be combined during the copy process.

NOTE:
Files can be copied to the same directory only if they are copied with a new name. If you copy a file to a different directory without specifying a new name, the file will be copied with the same name. If you attempt to copy a file to the same directory without providing a new name, DOS will cancel the copy and display the message:

File cannot be copied onto itself

Options

/A Used to copy ASCII files. Applies to the filename preceding it and to all following filenames. Files will be copied until an end-of-file mark is encountered in the file being copied. If an end-of-file mark is encountered in the file, the rest of the file is not copied. DOS will append an end-of-file mark at the end of the copied file.

/B Used to copy binary files. Applies to the filename preceding it and to all following filenames. Copied files will be read by size (according to the number of bytes indicated in the file's directory listing). An end-of-file mark is not placed at the end of the copied file.

/V Checks after the copy to assure that a file was copied correctly. If the copy cannot be verified, the program will display an error message. Using this option will result in a slower copying process.

Examples

The first filename you enter is the source file; the second file is the target file. To copy the file TEST.DOC from the current directory to drive B (with the same name), enter:

```
copy  test.doc  b:
```

To copy the file TEST.DOC to the current directory with the new name, "TEST2," enter:

```
copy  test.doc  test2
```

To copy and combine (concatenate) the files "TEST1.DOC" and "TEST2.DOC" to a new file, "TEST3," enter:

```
copy  test1.doc+test2.doc  b:test3
```

You can also combine files by using wildcard characters (? and *). To copy all files with a .DOC filename extension on drive C to a new file "ALLDOCS" on drive B, enter

```
copy  c:*.doc  b:alldocs
```

Other, more complicated, combinations are also possible while copying. For example, to combine all files with a .TXT filename extension with all files that have the same filename and a .DOC extension, copying the newly combined file to a new file on drive B with an .ADD extension, enter

```
copy  *.txt+*.doc  b:*.add
```

In this case, the file "TEST.TXT" will be combined with the file "TEST.DOC" resulting in a combined file with the filename "TEST.ADD."

CTTY

Type: Internal

Syntax:

CTTY (device)

Purpose: Changes the standard I/O (Input/Output) device to an auxiliary device.

Discussion

The CTTY command is used only for I/O (input/output) devices such as a monitor. It cannot be used for peripheral devices such as a printer. However, DOS does not check to make sure you have entered a valid device name. If, for example, you specify a printer, DOS will attempt to read from the device, producing an error condition.

Affects only programs that use standard DOS calls. BASIC and other programs that do not access I/O devices with DOS calls will not recognize device changes reset using this command.

Valid device names are:

AUX
CLOCK$
COM
CON
LPT
LST
NUL
PRN

Examples

To change to another I/O device using the AUX port, enter

ctty aux

To change back to the standard I/O device enter

ctty con

DATE

Type: Internal

Syntax:

DATE

DATE mm-dd-yy

Purpose: Displays and/or sets the system date.

Discussion

The date you set will be used by DOS to "date stamp" files that are created or modified. This date will be shown when you use the DIR command to display information about files.

You can specify the numbers to set the date at the same time you enter the DATE command or you can enter the command alone and the program will prompt you to enter the date.

For information on using the PROMPT command to display the date, refer to Chapter 5, "Tips for Advanced Users."

Example

When you enter

date

if the current date is set the program will display

```
Current date is Sat 06-11-1988
Enter new date (mm-dd-yy):
```

Press **Enter** to leave the date as it is, or type in the date using the format shown. Be sure to enter one or two numbers to indicate the month, then a hyphen, one or two numbers to indicate the day of the month, another hyphen, and one or two numbers to indicate the year. For example, to set the date on May 24, 1988, enter

5-24-88

DEL

Type: Internal

Syntax:

DEL (ERASE) [d:][path]filename

Purpose: Deletes (erases) files from disk.

Discussion

You can use wildcard characters (? and *) to delete groups of files. If you don't specify a pathname,, the program assumes the files to be deleted are on the drive and directory you are currently using.

CAUTION:

Be careful when you use this command with wildcard characters: groups of files can be erase with one command entry. For example, if you enter

del t*.*

all files in the current directory that begin with the letter "T" will be immediately erased.

Examples

To erase the file "LETTER3.TXT" from the "LETTERS" directory on the diskette in drive B, enter

del b:\letters\letter3.txt

To erase all the files in the current directory, enter

del *.*

If you use this form of the command (to delete all files in a directory), the program will display the prompt

Are you sure (Y/N)?

If you are sure you are deleting the files you want to delete, press the letter **Y** key to start the erasing process.

DIR

Type: Internal

Syntax:

DIR [d:][path][filename][/P][/W]

Purpose: Displays directory of files and directories stored on disk.

Discussion

In addition to files and directories, DIR also displays both the volume name and amount of free storage space on the disk (if there are files stored in the current directory). Note that both of these are for the entire DISK, not just for the path you specified.

The DIR command is also useful if you want to know what directories have been created on the specified disk. The directories will be displayed along with the files on the disk. They can be identified by the <DIR> label that follows the directory name.

Wildcard characters (? and *) can be used to specify groups of files.

For more information on using the DIR command, refer to Chapter 2, "Using Common DOS Commands."

When you use the DIR command, it will display all files that match the path and filename specifications, along with their size in bytes and the time and date of their last modification. If you don't enter one of the options, the entire list will be displayed, even if the list is too long to fit on one screen. To limit the amount of data displayed, use either of the two options described below.

Options

/P	Pauses when the screen is full. Press any key to display another screen full of data.
/W	Displays only filenames and directory names (without the added information about each file) in a five-wide display format.

Examples

To display only files with the ".TXT" filename extension on the current drive that begin with the letters "FIL" , enter

dir fil*.TXT

To display only files on drive C that have no filename extension, enter

dir c:*.

This form of the DIR command will also display directories. They can be identified by the <DIR> label that follows the directory name.

DISKCOMP

Type: External

Syntax:

DISKCOMP [d:] [d:][/1][/8]

Purpose: Compares the contents of two diskettes.

Discussion

DISKCOMP compares two disks, track by track, and displays the numbers of tracks that are not identical.

If you enter only one drive designation, DISKCOMP compares with the disk in the currently active drive.

If you enter one drive designation that is the same as the current drive (or if you don't enter a drive), the comparison will take place on one drive. DISCOMP will prompt you when it is time to insert each disk to be compared.

Disks copied using the DISKCOMP command should be reported to be identical when checked using this command; however, disks copied using the COPY command may not be identical since DOS may not copy files to the same locations on the disk.

Options

/1 Causes only the first side of the disk to be compared (even if the disk is double-sided).

/8 Causes only 8 sectors per track to be compared (even if the disk contains a different number of sectors per track).

If differences are found on a track, the program will display an explanatory message. For example, if track eight on side one are not the same, DOS will display

Compare error(s) on track 8, side 1

DISKCOPY

Type: External

Syntax:

DISKCOPY [d:] [d:][/1]

Purpose: Makes an exact copy of a diskette.

Discussion

This command is used only for copying diskettes, not fixed disks.

The first drive you specify is the drive for the source diskette. The second is the drive for the target diskette. DISKCOPY checks to determine if the disk in the target drive has been previously formatted. If not, DISKCOPY will format it before it starts the copy (except in early versions of DOS).

If the target drive is the same as the source drive (or if you do not enter a drive designator), the copying will be done using one drive. The program will prompt you when to insert each diskette.

Options

/1 Copies only the first side of the diskette, even if the target diskette is double sided.

After copying, the program will display

```
Disk copy OK
Copy another (Y/N) ?__
```

If you press the letter **Y** key, you will be prompted to insert another disk to copy using the same parameters. Press the letter **N** key to terminate DISKCOPY.

Example

To copy the contents of the disk in drive A to the disk in drive B (erasing any data already on the disk), enter

```
diskcopy a: b:
```

ECHO

Type: Internal

Syntax:

ECHO ON ¦ OFF
ECHO (message)

Purpose: Turns on or off the display of commands in a batch file. Can also be used to display a message in a batch file.

Discussion

Normally, commands in a batch file are displayed as the batch file runs. ECHO OFF turns off this feature. ECHO ON turns the display feature back on.

If you enter the ECHO command with no options, the program will display the current ECHO setting.

For more information on using the ECHO command and other batch commands, refer to Chapter 4, "Using Batch Files." A variety of examples are presented in that chapter and in Chapter 5, "Tips for Advanced Users."

Example

If you want to display the message "This is file one" during the execution of a batch file, but you do not want to display the commands being acted upon, enter

```
echo off
echo This is file one
```

Notice that you do not have to surround the message with quote marks as is required in BASIC and other programming languages.

EXE2BIN

Type: External

Syntax:

EXE2BIN [d:][path]filename [d:][path]filename

Purpose: Converts files from .EXE format to binary format. This results in files that require less disk space and may also result in a faster loading file.

Discussion

The first file specification you enter is to indicate the source file. If you don't enter a filename extension, the program assumes an .EXE extension.

The second file specification you enter is to indicate the new file that will be created using the .BIN format (sometimes referred to as a "memory image" of the program). If you do not enter a filename, EXE2BIN will use the same filename as the source file. If you do not specify a filename extension, the program will assign the .BIN extension.

The input file must be in valid .EXE format as produced by the MS-DOS file linker. The "resident," or actual code and data part of the file, must be less than 64K bytes. There must be no STACK segment. Two kinds of conversion are possible depending on the specified initial CS:IP:

- If CS:IP is not specified, a pure binary conversion is assumed. If segment fixups are necessary (the source program contains instructions that require segment relocation), the program will display

  Fixups needed - base segment (hex):

  Execution may be continued by typing a legal hexadecimal number and pressing the **Enter** key.

- If CS:IP is specified as 100H, EXE2BIN assumes the file will run as a .COM file with the location pointer set at 100H by the assembler statement ORG (the first 100H bytes of the file are deleted). In this case, no segment fixups are allowed, since .COM files must be segment relocatable.

Example

To convert the file "CHECK.EXE" to binary format and rename the file "CHECK2.BIN, enter

exe2bin check.exe check2.bin

EXIT

Type: Internal

Syntax:

EXIT

Purpose: Exits a secondary command processor.

Discussion

This command can be used when you are running an application program and want to start the DOS command processor, then return to your program. For example, to look at a directory on drive B while running an application program, you must start the command processor by typing COMMAND in response to the default drive prompt.

You can now type the DIR command and DOS will display the directory for the default disk. When you type EXIT, you return to the previous level (your application program).

Example

If you start a new command processor by entering

command /c

then you can return to the previous command processor by entering

exit

FASTOPEN

Type: External

Syntax:

FASTOPEN d:[=n]

Purpose: Keeps track of the locations of files for fast access.

Discussion

FASTOPEN locates files on a fixed disk by searching your disk's directory structure in a predetermined search pattern. After running FASTOPEN, each time you use a file or directory, the program will make a record (in memory) of the file's name and location. The next time you specify that file, DOS will look first in the last known location. This reduces the time needed to access files.

This command is used only on fixed disks and can be used on up to four fixed disks at one time. For each fixed disk, you enter the number of files you want to keep track of (from 10 to 999 files). If you do not enter a number, DOS will keep track of 10 files. FASTOPEN uses about 40 bytes of memory for each file or directory location it keeps track of.

You can use this command to set the number of files to track only once. If you want to change the FASTOPEN settings, restart your computer.

Example

To keep track of 30 files on drive C, enter

fastopen c:=30

FC

Type: External

Syntax:

FC [/A][/C][/L][/Lb n][/N][/T][/W][/(number)] [d:][path]filename [d:][path]filename

or (for binary comparisons)

FC [/B][/number] [d:][path]filename [d:][path]filename

Purpose: Displays the differences between two files or sets of files.

Discussion

FC reports differences between the two files you specify. FC first displays the name of the first file to compare. Next, it displays the lines that differ between the two files. Next, it displays the first line that matches in the two files. FC next displays the name of the second file, followed by the lines that are different followed by the first line that matches.

The FC command requires enough system memory (RAM) to compare 100 lines. If system memory is limited, FC will make comparisons between the number of lines that can fit into available memory. Be aware that this can change the results of a comparison.

Options

/A Instead of displaying all the lines that do not match (in an ASCII comparison), the program displays only the lines that begin and end each set of differences.

/B Used for a binary comparison of files. The files are compared byte-by-byte. Mismatches are displayed as a relative address from the beginning of the file. Unless you specify the /L switch, the program will automatically assume you want to use this type of comparison when the compared files have a .COM, .EXE, .SYS, .BIN, .OBJ, or .LIB filename extension.

/C Ignores the case of letters in the compared files. All letters are considered as upper case.

/L Used for an ASCII comparison of files. Unless you specify the /B switch, the program will automatically assume you want to use this type of comparison for all files that do not have a .COM, .EXE, .SYS, .BIN, .OBJ, or .LIB filename extension.

/Lb n Sets the line buffer to n number of lines. The amount of memory used for holding lines being compared is normally enough to store about 100 lines. Use this option to reset the buffer (memory space) to n lines. If more than this number of consecutive lines is found to differ, the program will terminate the comparison and display an error message.

/N Used to turn on the display of line numbers while carrying out an ASCII comparison.

/T Turn off the display of tabs as spaces in an eight column display format.

/W Compresses tabs and spaces during the comparison.

/(number) The number of lines that must match after the program finds a difference between the compared files (if fewer lines match, the matching lines are displayed as differences). If you do not use this option, the number of lines to match between the files is two.

Example

To compare the text file LETTER.TXT with the file MEMO2.TXT on drive B (an ASCII comparison), enter

```
fc  /l  letter.txt  b:memo2.txt
```

To carry out a binary comparison of the files CHECK.EXE and CHECK2.EXE, enter

```
fc  /b check.exe  check2.exe
```

FDISK

Type: External

Syntax:

FDISK

Purpose: Prepares a fixed disk to accept DOS files for storage.

Discussion

FDISK provides a set of options related to the preparation of a fixed disk.

> **NOTE:**
> This command is implemented in different ways by different manufacturers. Some may replace it with a command of a different name that carries out somewhat different functions.

A menu will usually be displayed offering these FDISK functions:

1. Create DOS partitions on the fixed disk. Using this option, you can create more than one logical drive on one physical fixed disk.

2. Change an existing partition on the fixed disk.

3. Delete an existing partition on the fixed disk.

4. Display current partition information.

The program will display a series of questions. Your responses indicate the configuration options for the fixed disk.

For more information on using FDISK and other fixed-disk commands, refer to Chapter 3, "Using Fixed Disks."

FIND

Type: External

Syntax:

FIND [/V][/C][/n] "string" [d:][path]filename[...]

Purpose: Finds and reports the location of a specific string of text characters in one or more files.

Discussion

FIND is a "filter" command (reads from input, transforms it, and outputs it to the screen, to a file, or to a printer). FIND searches for a string of characters you enter in the files you name. The program will display the lines that contain the specified string. You must put double quote marks before and after the characters to be searched for. If the string you are searching for already has quotes around it, you must use a double set of quote marks.

If you do not enter a filename, the program will prompt you for one when the program starts. Wild card characters are not allowed.

For more information on using FIND and other DOS filter commands, refer to Chapter 5, "Tips for Advanced Users."

Options

/V Displays all lines not containing the specified string.

/C Displays only the count of the number of lines that contained a match in each of the files.

/N Displays the found line preceded by the relative line number.

Example

You can use FIND to search more than one file. For example, to find the string "and being unknown, is Truth" in both the file QUOTES1.TXT and QUOTES2.TXT on the diskette in drive B, enter

```
find  "and being unknown, is Truth"  b:quotes1.txt b:quotes2.txt
```

FIND will display the found lines first from the file "QUOTES1.TXT" and then from the file "QUOTES.TXT."

FOR

Type: Internal

Syntax:

FOR %%(variable) IN set DO (command)

Purpose: Performs repeated execution of commands (for both batch processing and interactive processing).

Discussion

%%variable can be any character.

set can be one or more file specifications.

The **%%variable** is set sequentially to each of the set entries. Then the DOS command specified after the DO is carried out. Typically **set** will include wildcard characters (? and *) to indicate a group of filenames. Then, **%%variable** is set to each matching filename. You can also use path designations for **set.**

This command provides a way to repeat DOS commands in a predetermined way (usually to tell DOS to carry out a command on each of the files named in the set specifications).

Any character except 0 through 9 can be used for **%%variable** (to avoid confusion with the %0-%9 batch parameters).

Examples

To TYPE (display) the contents of every file in the current directory that has the filename extension .TXT, enter

```
for %%X in (*.txt) do type %%X
```

If the program finds files on the disk that have the TXT extension, the command sets %%X to each of the .TXT files then displays the contents of each file.

FORMAT

Type: External

Syntax:

FORMAT d:[/1][/2][/4][/8][/n:(sectors)][/n:(tracks)][/B][/S][/V]

Purpose: Formats a disk to accept DOS files.

Discussion

Formats the disk in the specified drive to accept DOS files, analyzing the entire disk for defects.

Initializes the directory and file allocation tables. Can be used to format both diskettes and fixed disks.

> **NOTE:**
> In some earlier version of DOS, the drive designation letter was optional. If you are using one of these versions, you can format a diskette or a FIXED DISK if you enter
>
> **format**
>
> while working in that drive. The section named "Advanced Batch Files" in Chapter 4, "Using Batch Files" describes a batch file that can protect against accidental formatting of the fixed disk.

Options

/1 Format for single-sided use, regardless of the drive type.

/2 Format for double-sided use, regardless of the drive type.

/4 Format a double-density diskette in a high-density type disk drive. Files written to a double sided disk using a high-density drive may not be reliable.

/8 Format at 8 sectors per track. If /8 is not specified, FORMAT defaults to 9 or 15 sectors per track, depending upon the disk drive type. The /V option cannot be used with the /8 option.

/n:(sectors) Specifies the number of sectors per track on the disk. Used to format a 3.5 inch disk with the number of sectors per track specified using this option. For 720 K-byte disks, this value should be entered as "N:9."

/n:(tracks) Specifies the number of tracks on the disk. Used to format a 3.5 inch disk with the number of tracks specified using this option. For both 720 K-byte disks and 1.44 K-byte disks, this value should be entered as "T:80."

/B Formats a disk reserving room on the disk to later copy the DOS system file.

/V Causes FORMAT to prompt for a volume label after the disk is formatted. The label can be of 1 to 11 characters. The same characters acceptable in filenames are acceptable in the volume label (however, you cannot add a filename extension). The /8 option cannot be used with the /V option.

/S Copies the operating system files to the disk after formatting. These system files are hidden files and will not appear in a directory listing. Using some versions of DOS, this must be the last option entered.

GOTO

Type: Internal

Syntax:

GOTO (label)

Purpose: Causes unconditional branch to the specified label.

Discussion

Sends the program to the command found in the line immediately after the label definition.

A label is the first eight characters on a batch file line after a colon. The label can be indented, as long as there are only spaces preceding it on that line. The label (the string of eight characters) can include spaces but not other restricted characters such as commas, semicolons, or equal signs.

A label name after a GOTO command transfers processing to the line containing the label. If a label is not found (or if the label contains illegal characters), an error message is displayed and the batch file terminates.

For more information on using the GOTO command and other batch commands, refer to Chapter 4, "Using Batch Files."

Example

To create a batch file that will function as an infinite loop displaying the message "This is an infinite loop," enter

```
:START
echo  This is an infinite loop goto START
```

A number of more useful examples using the GOTO command can be found in this chapter.

GRAFTABL

Type: External

Syntax:

GRAFTABL (code page)
GRAFTABL /STATUS

Purpose: Loads a table of character data into memory (for use with a color/graphics adapter).

Discussion

Used to load the table of displayable characters to those appropriate for the specified country (when in graphics mode using a color/graphics adapter). After using this command, the ASCII characters 128 through 255 can be displayed in graphics mode with a color/graphics adapter.

The normal code page is set for the United States with versions of MS-DOS sold in the U.S.

Using the GRAFTABL command reduces the amount of available memory by 1360 bytes.

Using the GRAFTABL command resets the ERRORLEVEL code according to the following table:

0	Code page now resident; no previous code page was set.
1	If a new code page was specified, it is now resident. There was a code page table previously loaded; it was replaced by the current request.
2	No code page table was previously loaded; no new table loaded.
3	Incorrect parameter; no new table loaded.
4	Incorrect version of DOS; DOS version 3.30 required.

Options

code page The available code pages and related countries are:

437 United States
860 Portugal
863 Canada (French)
865 Norway and Denmark

/STATUS Displays the number of the currently selected code page.

Example

To load the GRAFTABL for Norway and Denmark, enter

```
graftabl 865
```

GRAPHICS (Screen Print)

Type: External

Syntax:

GRAPHICS [printer type][/B][/R][/LCD][/p=port]

Purpose: Provides a way to print contents of a graphics screen display.

Discussion

After using this command, pressing **Shift** + **PrtSc** (or calling for a printout using some application programs) prints the screen contents on the printer in graphics mode (printer must be capable of printing in graphics mode). If the display is currently in 320 by 200 color/graphics mode, and if you are using a graphic printer, the screen will be printed with up to four shades of gray. If the display is currently in 640 by 200 color/graphics mode, the screen contents will be printed sideways.

You can also print the contents of the screen from a program, use the following coding:

```
PUSH BP
INT 5
POP BP
```

In graphics mode, printing will take considerably longer.

Options

printer type The printer capabilities will change with different computer-/printer combinations; however, the setting for this option will generally be as follows:

COLOR1	Standard graphics printer
COLOR4	Graphics printer with standard four color ribbon
COLOR8	Graphics printer with special four color ribbon
COMPACT	Compact graphics printer

GRAPHICS	Dedicated graphics printer
THERMAL	Thermal paper printer

/B Prints the background in color using a graphics printer with a color ribbon (use /COLOR4 or /COLOR8 option as the printer option).

/R Reverses the print output to print white on black (as seen on the computer screen).

/LCD Prints from a LCD-type of computer screen.

P=port Sets the printer port used when you press the **Shift** + **PrtSc** key combination (1, 2, or 3). If you do not specify this option, DOS will use port #1.

IF

Type: Internal

Syntax:

IF [NOT] exist (filename) (command)

IF [NOT] string1==string2 (command)

IF [NOT] ERRORLEVEL (number) (command)

Purpose: Allows for conditional operations in batch processing.

Discussion

When used in combination with the GOTO command, the IF command provides a way to transfer control within a batch file dependent on the outcome of a test.

For more information on using the IF command and other batch commands, refer to Chapter 4, "Using Batch Files."

The IF test is indicated by the "condition" label in the command format. It can be one of three types:

Condition 1: IF [NOT] EXIST (filename) (command)

This condition is determined by a test to see if a file exists (or does not exist) on disk.

When this conditional test is included (it can be on any line of your batch file), DOS checks to determine if the specified file exists (you must specify the path so DOS can find the file). If the condition is true (as stated), the specified command is executed. If the condition is found to be false, the command is not acted on and processing proceeds to the next line in the batch file.

When the IF command is used together with the GOTO command, it provides a way to transfer control to a different line in the batch file dependent on the outcome of the test.

For example, to display a warning message only if the file "DATA.1" does not exist in the current directory, enter

```
echo off
if not exist data.1  goto   MESSAGE
echo  The file DATA.1 exists on the current directory
goto  END
MESSAGE
echo  The file DATA.1 was not found.
echo  The file must be in the current directory
:END
```

When you enter the name of this batch file, there are two possible outcomes. If the file DOES NOT EXIST in the current directory, the program will display:

```
echo off
The file DATA.1 was not found.
The file must be on the current directory
```

If the file DOES EXIST in the current directory, the program will display:

```
echo off
echo  The file DATA.1 exists on the current directory.
```

In this example, the IF command is combined with the GOTO command to determine the optional branching. The command tells DOS to check the current disk to determine if the file DATA.1 exists. Since we are using the NOT EXIST form of the command, the condition is true if it DOES NOT find the file; since the condition is true, the specified command (in this case, the GOTO command) is acted on. The "GOTO MESSAGE" line passes processing to the ":MESSAGE" label and the messages on the following lines are displayed.

Condition 2: IF [NOT] string1 = = string2 (command)

This operation checks to determine if the first string you enter is equal (or is NOT equal) to the second string you enter. If the condition is true (as stated), the specified command will be carried out.

DOS judges this condition based on whether or not the two strings are EXACTLY identical. DOS WILL detect a difference between uppercase and lowercase letters.

You can create a very useful version of the IF command that checks to determine whether of not a parameter was passed to the batch file from the command line.

The batch file below acts conditionally depending on a test of whether or not any characters were entered on the command line after the batch file name.

```
echo off
if not x==%1x goto  PRINT
echo  You must enter a parameter immediately after batch file name
goto  END
:PRINT
echo  The passed parameter is [ %1 ]
:END
```

The first line of this batch file checks to determine whether or not a parameter has been passed at the command line. It uses the IF command to determine if string one (the character 'x') is equal to string two (the character 'x' plus the %1 symbol that represents the first passed parameter). If a parameter WAS passed to the batch file, the two strings will not match and the condition, (stated as NOT equal) is true; the GOTO command will be acted on, transferring processing to the label ":PRINT."

If the condition is found to be false (the two strings ARE equal), it means a parameter WAS NOT entered on the command line. In that case, the GOTO command on the same line as the IF command will not be acted on and processing will be transferred to the next line of the batch file. The next line includes the message "You must enter a parameter immediately after the batch file name." The following line transfers the processing to the ":END" label and the program terminates.

If characters WERE entered after the batch filename, the program will display

```
echo off
The passed parameter is [ test ]
```

If characters WERE NOT entered after the batch filename, the program will display

```
echo off
You must enter a parameter immediately after batch file name
```

Condition 3: IF [NOT] ERRORLEVEL (number) (command)

This operation checks to determine if an ERRORLEVEL number is set (or is NOT set) at or greater than the specified value. The ERRORLEVEL number is set by programs and the value is retained in the computer's memory. This command is used to check that value. DOS judges this condition to be true if the ERRORLEVEL code is equal to OR GREATER than the number you specify.

A common use of the IF command is to display a message to indicate that an ERRORLEVEL has been set. To display a message whenever any ERRORLEVEL number has been set (any value above zero), enter:

```
echo off
if errorlevel 1 echo The errorlevel is now greater than zero
```

When this line is acted on (and if the ERRORLEVEL is found to be set to a value), the ECHO command will be acted on and the program will display

```
echo off
The errorlevel is now greater than zero
```

If DOS does not find a value set for ERRORLEVEL, the program terminates without displaying the message.

JOIN

Type: External

Syntax:

JOIN d: d: path

JOIN d:/D

Purpose: Allows access to the directory structure and files of a drive through a directory on a different drive.

Discussion

After joining a drive to a directory, the files stored on the joined drives can be accessed as if they were on the specified directory.

Attempts to access the joined drive will result in the display of the error message

Invalid drive specification

If you try to join a directory containing files, the program will display the error message

Directory not empty

You cannot join the current directory. Also, you cannot join to a directory that is more than one level below a root directory or to the root directory itself.

If the directory you specify in the second parameter, does not exist, DOS will create it. If the directory does exist, it must be empty.

For more information on using JOIN and other fixed-disk commands, refer to Chapter 3, "Using Fixed Disks."

Option

/D Used to disconnect a joined drive.

Example

To join drive B to the directory "FILES" on drive A, enter

```
join  b:  a:\files
```

If you now ask for a directory of A:\FILES, the files stored on drive B will be displayed.

After entering this command, attempts to access drive B will result in the error message

```
Invalid drive specification
```

To disconnect the join established in the previous example, enter

```
join  b:  /d
```

KEYB

Type: External

Syntax:

KEYB

KEYB [xx][,][yyy][,][d:][path]filename

Purpose: Loads a program that replaces the support program for U. S. keyboards.

Discussion

The two letter code, **xx** must be one of the following:

US	UNITED STATES
FR	FRANCE
GR	GERMANY
IT	ITALY
SP	SPAIN
UK	UNITED KINGDOM
PO	PORTUGAL
SG	SWISS-GERMAN
SF	SWISS-FRENCH
DK	DENMARK
BE	BELGIUM
NL	NETHERLANDS
NO	NORWAY
LA	LATIN AMERICA
SV	SWEDEN
SU	FINLAND

The three letter code, **yyy** represents the code page that defines the character set (refer to the GRAFTABL command for more information on code page usage).

The filename you specify is the keyboard definition file.

If you use this command without options, the program will display the current keyboard code setting, the related code page, and the current code page setting.

You can change from the current KEYB setting to the United States keyboard format by pressing and holding the **Ctrl** key while pressing the **Alt** and **F1** keys simultaneously. Return to the previous KEYB setting by pressing and holding the **Ctrl** key while pressing the **Alt** and **F2** keys simultaneously.

Using the KEYB command resets the ERRORLEVEL code according to the following table:

0	Successful execution
1	Invalid language, code page, or incorrect syntax
2	Bad or missing keyboard definition file
3	The keyboard table could not be created
4	An error was encountered during communication with the COM device
5	The code page has not been prepared
6	The translation table for the selected code page could not be found in the keyboard table

Example

To use the Spain keyboard, enter

```
keyb sp
```

LABEL

Type: External

Syntax:

LABEL [d:][volume label]

Purpose: Creates or changes or deletes a volume label for a disk.

Discussion

A volume label may consist of up to eleven characters (the same characters that are legal for a file name). If you enter an erroneous volume label, the error message "Invalid volume label" appears, and the system prompts for a re-entry.

If you use this command without entering a volume label, the program displays

Input volume label - Up to 11 characters.
Press ENTER only to delete label.

Enter a volume label, or press the **Enter** key to delete the current volume label. To display the current label, use either the DIR or VOL command.

Example

To create the label "TESTVOL1-A" on the diskette in drive B, enter

label b:testvol1-a

MKDIR (MD)

Type: Internal

Syntax:

MKDIR (MD) [d:]path

Purpose: Creates a new subdirectory.

Discussion

If you do not specifically enter a path designation, the directory will be created as a subdirectory within the current directory.

There is no limit to the number of directories you can create. However, the maximum length of any single path specification is 63 characters, including backslashes.

For more information on using MKDIR and other fixed-disk commands, refer to Chapter 3, "Using Fixed Disks."

Examples

To create the new subdirectory "LETTERS" within the existing directory "NEWSALES" on drive A, enter

md a:\newsales\letters

If you are already working in the "NEWSALES" directory, you can create the new subdirectory "LETTERS" within that directory by entering

md letters

MODE

Type: External

Syntax:

MODE n

MODE LPT#[:][n][,][m][,][P]

MODE [n],m[,T]

MODE COMn[:]baud[,][parity][,][databits][,][stopbits][,][P]

MODE LPTn[:]=COMn

MODE (device) CODEPAGE PREPARE=(codepage)
[d:][path]filename

MODE (device) CODEPAGE PREPARE=(codepage list)
[d:][path]filename

MODE (device) CODEPAGE SELECT=(codepage)

MODE (device) CODEPAGE [/STATUS]

MODE (device) CODEPAGE REFRESH

Purpose

MODE sets the mode of operation for devices or communications. It can be used to set the mode for printers, monitors, or for the serial interface. It can be used to prepare and select code pages and to redirect printer output to the serial interface.

The MODE command can be used for four different types of operations. Each is summarized below.

Option 1. Printer Settings

Syntax:

MODE LPT#[:][n][,][m][,][P]

Discussion

Use this version of the MODE command to set the output to the printer as either 80 or 132 characters per line. Unless this option is specifically set to 132, DOS uses the 80 characters per line setting for all printouts.

When any one of the DOS commands is used to send lines to the printer, this setting determines the MAXIMUM number of characters that will be sent before a line feed is sent to the printer to force the printing to continue on the next line.

Options

This option sets the printer number. You can enter 1, 2, or 3 to designate one of three printers.

n characters per line (80 or 132)

m number of lines per inch (6 or 8) This option sets the output to the printer as either 6 or 8 lines per inch. Unless this option is specifically set to 8, DOS uses the 6 lines per inch setting for all printouts.

When using the MODE command to set either the number of characters per line or the number of lines per inch, the program will not warn you if you enter an illegal number.

P places a portion of the MODE program in memory and tells DOS to continuously try to send output to the printer if a time-out error occurs. This gives you a way to correct a printer error without terminating the printout. An example is when the printer returns an out-of-paper error. If you have used the P option, the computer will continue to try to send while you refill the paper supply. The printing will continue when the out-of-paper condition is resolved.

If an error occurs that can't be so easily resolved, you can stop the continuous retrying by pressing and holding the **Ctrl** key and then pressing the **Break** key.

To change this condition, you must re-enter the MODE command with the printer options, but without using this P option.

Example

MODE LPT1,132,8

Sets printer number 1 to 132 characters per line and 8 lines per inch.

Option 2. Monitor Settings

Syntax:

MODE n

MODE [n],m[,T]

Options

n should be one of the following expressions:

40	40 characters per line
80	80 characters per line
BW40	40 characters per line, black and white (color display adapter)
BW80	80 characters per line, black and white (color display adapter)
CO40	40 characters per line, color (color display adapter)
CO80	80 characters per line, color (color display adapter)
MONO	monochrome display adapter
m	R or L (shifts display right or left)
T	Displays a test pattern used to align the display.

Discussion

Most computers include a switch on the system board that specifies the type of monitor you will be using. Other computers set the monitor display options using a special software program. Either way, the MODE command can usually be used to override this setting. If you change monitors, you can quickly reset the output format used by your display adapter by using this command.

You can shift the display left or right one character (40-character mode) or two characters (80-character mode) by specifying **R** or **L**.

When you use the T option, the program will display a prompt that asks if the screen is aligned properly. If you enter **Y**, the command ends. If you enter **N**, the shift is repeated, followed by the same prompt.

Examples

If you enter

mode 80,r,t

The display mode is set to 80 characters-per-line and the display is shifted two character positions to the right. The test pattern is displayed so that you can shift the display again without re-entering the command.

If your monitor display needs aligning, you may want to put a form of this command in your AUTOEXEC.BAT file. If your display always needs to be shifted to the right, enter the following line into your AUTOEXEC.BAT file.

c:\dos\mode ,r

Each time this command is issued, the display will be shifted two characters to the right. Remember, MODE is an external DOS command. To use it in your AUTOEXEC.BAT file you must tell DOS where to find the MODE.COM program (in this case, in the DOS directory.

Option 3. Asynchronous Communications Settings

Syntax:

MODE COMn[:]baud[,][parity][,][databits][,][stopbits][,][P]

Options

n 1 to 4 to indicate the asynchronous communications controller number (the communications port you want to use for communications). This will generally be set at 1 (unless your computer uses more than one output port).

baud 11 (110 baud), 15 (150 baud), 30 (300 baud), 60 (600 baud), 12 (1200 baud), 24 (2400 baud), 48 (4800 baud), or 96 (9600 baud)

parity N (none), O (odd), or E (even) (default = E)

databits 7 or 8 (default = 7)

stopbits 1 or 2 (for 110 baud, default is 2; for all other baud rates, default is 1)

P specifies continuous retry option for time-out errors

> **NOTE:**
> The **parity, databits,** and **stopbits** settings depend on the control parameters required by your communications software.

Discussion

These parameters are used to initialize the asynchronous communications controller. The baud rate is required, but for the other parameters, you may enter a comma to accept the default value.

As discussed above concerning the MODE option for printers, the **P** option tells DOS to continuously try to send output to the printer if a time-out error occurs. This gives you a way to correct a printer error without terminating the printout. You can stop the continuous retrying by pressing and holding the **Ctrl** key and then pressing the **Break** key.

The **P** option is generally used when the asynchronous controller is being used with a serial interface printer.

Different hardware manufacturers may provide slightly different versions of this command.

Examples

To set the mode of operation to 1200 baud, no parity, eight databits, and one stopbit, enter

```
mode  com1:12,n,8,1,p
```

To accept the default values (even parity, 7 databits, and 1 stopbit) and use the "P" option, enter:

```
MODE  COM1:12,,,,P
```

Option 4. Direct Parallel Printer Output (to an asynchronous communications controller)

Syntax:

MODE LPT#[:]=COM n

Options

The printer number (1 to 4)

n The asynchronous communications controller number (1 to 4)

Discussion

This command redirects output directed to printer LPT# to asynchronous controller n.

You have to initialize the asynchronous controller with option 3 when you use option 4. You also have to specify the P parameter when a serial interface printer is connected to the RS-232C controller.

Option 5. To set MODE for Code Pages

Syntax:

MODE (device) CODEPAGE

Used to display the current code page.

MODE (device) CODEPAGE REFRESH

Used to refresh the active current code page when the resident code page settings for a device have been lost. This can occur, for example, when a code page has been set for a printer and the printer loses power. The words "CODEPAGE REFRESH" can be abbreviated as "CP REFRESH."

MODE (device) CODEPAGE PREPARE=((codepage list) [d:][path]filename)

MODE (device) CODEPAGE PREPARE=((codepage) [d:][path]filename)

Used to specify the prepared code page for the named device. The words "CODEPAGE PREPARE" can be abbreviated as "CP PREP."

MODE (device) CODEPAGE SELECT=(codepage)

Used to select the code page to be activated for the specified device. The words "CODEPAGE SELECT" can be abbreviated as "CP SELECT."

Options

device	Used to specify one of these device identifiers: CON, PRN, LPT1, LPT2, LPT3.
codepage	Used to specify the code page number (refer to Appendix B, "Country-Specific Information" for information on these code page numbers.
codepage list	Used to specify the list of code page numbers (refer to Appendix B, "Country-Specific Information" for information on these code page numbers.

Discussion

The country-specific options are new to DOS version 3.3.. They are used to pass country-specific information to DOS. They offer a number of new files with functions related to the use of languages. Some of these files have a ".CPI" filename extension; it is these filenames that are specified when using the country- specific options of the MODE command described in this section.

Examples

To display the currently active code page for the device LPT1, enter

mode lpt1 cp codepage

To re-establish a previously set code page for the device LPT1, enter

mode lpt1 cp refresh

To specify three different code pages related to the CONSOLE device using the file "EGA.CPI," enter

```
mode  con  cp prep=((860,,865)  EGA)
```

Notice that the comma (without entering a code page number) is used to specify that the second code page previously prepared does not change.

To select the code page 850 for the printer LPT1, enter

```
mode  lpt1  cp  select=850
```

The specified code page must be one of the code pages previously specified with CODEPAGE PREPARE.

MORE

Type: External

Syntax:

MORE < (filename or command)
(name) | MORE

Purpose: Sends output to console, one screen at a time.

Discussion

MORE is a "filter" command (reads from input, transforms it, and outputs it to the screen, to a file, or to a printer). MORE displays the output one screen at a time, then pauses and displays the message "--MORE--" at the bottom of your screen.

Pressing any key will display another screen full of information. This process continues until all the input data has been read. The MORE command is useful for viewing a long file one screen at a time.

You can use the MORE command with the redirection symbol (<) to send the output to a file or for use by another command.

You can also use the piping symbol (|) to pipe the output of a program or a DOS command (referred to as "name" in the command format) to MORE.

For more information about redirection and piping, refer to the section by that name in Chapter 5, "Tips for Advanced Users."

Examples

To display the file "SALE2.TXT," one screen full of data at a time, enter

more < sale2.txt

To display the output of the TYPE command one screen at a time, you can enter the command using the following format:

TYPE (filename) | MORE

For example, to display the file "LET3.DOC" that is in the "LETTERS" directory on drive B, enter

type b:\letters\let3.doc | more

NLSFUNC

Type: External

Syntax:

NLSFUNC [d:][path]filename

Purpose: Used to load a file with country-specific information.

Discussion

NLSFUNC, new to DOS with version 3.3, supports the use of extended country-related information. The filename you enter should contain the country-specific information (it is defined by a COUNTRY command placed in your CONFIG.SYS file in the root directory of your startup disk. If DOS does not find a COUNTRY command in the CONFIG.SYS file, it will look for a COUNTRY.SYS file in the root directory.

NLSFUNC must be run before you can use the CHCP command.

Example

To use the information found in the file "COUNTRY.SYS," enter

nlsfunc

To use the information found in the file "SPECIAL.SYS," enter

nlsfunc special.sys

PATH

Type: Internal

Syntax:

PATH [d:]path[;][d:]path[...]

Purpose: Sets or displays directories that will be searched for programs not in the current directory.

Discussion

PATH tells DOS which directories should be searched for external commands after DOS searches your working directory.

DOS searches the paths in the order specified in the PATH command.

For more information on using the PATH command and other commands related to using fixed disks, refer to Chapter 3, "Using Fixed Disks."

If you enter the PATH command without options, the program will display the currently set path designations.

Examples

To tell DOS to search for external DOS commands and other executable programs in the "PROGRAM" directory that is within the root directory on drive C, enter

```
path c:\program
```

You can tell DOS to search more than one path by specifying several pathnames separated by semicolons. For example:

```
path \program;\files\sales;\data2
```

To reset the path designation so that DOS will search only the current directory, enter

```
path ;
```

PAUSE

Type: Internal

Syntax:

PAUSE [comment]

Purpose: Suspends execution of a batch file until a key is pressed.

Discussion

PAUSE suspends execution of the batch file and displays the message

Strike a key when ready ...

With some versions of DOS, a comment may also be entered on the same line as PAUSE. The comment prompt is displayed before the "Strike a key when ready" message.

Example

To suspend execution of a batch file and display the message "Insert the target diskette in drive A," enter

pause Insert the target diskette in drive A

PRINT

Type: External

Syntax:

**PRINT [/B:(buffersize)][/D:(device)][/M:(maxtick)][/Q:(value]
[/S:(timeslice)][/U:(busytick)][/C][/P][/T] [d:][path][filename] [...]**

Purpose: Queues and prints data files.

Discussion

Using the PRINT command, you can set up one or more files to print and continue to use other programs (often referred to as "background" printing).

Options

/C Provides a way to select which files to cancel. A filename already in the print queue that precedes this option will be suspended. Names of files (already in the print queue) that follow this option will be canceled. You can also list more files to print as part of the same PRINT command by entering the /P option followed by the filenames to print.

/P Sets the print mode. The preceding filename and all following filenames will be added to the print queue. You can also list files to cancel as part of the same PRINT command by entering the /C option followed by the filenames to cancel.

/T Deletes all files in the print queue (those waiting to be printed). After the print files are canceled, the program displays a cancellation message. If a file is being printed as the PRINT command is used with this option, the printing will stop immediately.

The following options are allowed only the first time you run the PRINT command:

/D (device) Specifies the print device. If not specified, PRINT will prompt you to enter the name of a print device. The default is PRN.

/B (buffersize) This parameter sets the size (in bytes) of the amount of memory used by the PRINT buffer. Increasing this value speeds up the operation of the PRINT command. If you do not use the /B option, the PRINT buffer will be set to 512 bytes.

/M (maxtick) Specifies how many computer clock ticks PRINT can have to print a file. The value of /M can be from 1 to 255. If you do not use the /M option, this parameter will be set to 2.

/Q (value) Specifies the maximum number of files that are allowed in the print queue (from 4 to 32). If you do not use the /Q option, this parameter will be set to 8.

/S (timeslice) Specifies the time slice (the interval of time used by DOS to schedule PRINT jobs. If you do not use the /S option, this parameter will be set to 8.

/U (busytick) Specifies the number of computer clock ticks that DOS will wait for a printer that is not available. If the wait is longer than this value, DOS gives up its time slice (refer to option /S above). If you do not use the /U option, this parameter will be set to 1.

If you use the PRINT command without entering any options, the current contents of the print queue is displayed. This does not change the status of the files currently in the PRINT queue. If you do not enter any of the forward slash options (/), the /P option is assumed.

The first time you use the PRINT command, the following prompt appears (unless you specify the /D parameter):

Name of list device [PRN:]

Whatever device you enter becomes the PRINT output device. Press the **Enter** key to accept the default (PRN).

During use of the PRINT command, one or more messages may be displayed. The message and its meaning are summarized below.

Message: List output is not assigned to a device

Meaning: The name of the printer you specified was not valid.

Message: PRINT queue is full

Meaning: You attempted to enter more than 10 files in the PRINT queue.

Message: PRINT queue is empty

Meaning: There are no files in the print queue.

Message: No files match d:(filename)

Meaning: No files match the filename specification you entered.

Message: All files canceled by operator

Meaning: You entered the /T (terminate) option to cancel all the files currently in the PRINT queue.

Message: File canceled by operator

Meaning: You entered the /C (cancel) option to cancel the file currently in the PRINT queue.

Examples

To stop printing and cancel the print queue, enter

```
print /t
```

To remove the files "LETTER.TXT, "START.EXE," and "RATE.COM" (all on drive B) from the PRINT queue, enter

```
print b:letter.txt/c b:start.exe b:rate.com
```

To remove the file "SALES.DOC" from the PRINT queue and, at the same time, add the files "OLDSALE.DOC" and "NEWSALE.DOC," enter

```
print sales.doc/c oldsale.doc/p newsale.doc
```

PROMPT

Type: Internal

Syntax:

PROMPT [prompt text][options]

Purpose: Changes the DOS command prompt.

Discussion

You can change the DOS prompt (the indicator that DOS is ready for input) to almost any type of special prompt you want. For example, you can use this command to make the prompt display the current time, the date, or the current directory. You can also use combinations of the special prompt characters to create custom prompts.

For more information on the many uses of the PROMPT command, refer to Chapter 5, "Tips for Advanced Users."

If you use the PROMPT command with no options, the prompt will be set to the default prompt (the current drive letter and a greater-than symbol).

By using the characters indicated in the following list, you can create a great variety of prompts (each must be preceded by a dollar sign):

Character	Prompt
$	'$' character
t	current time
d	current date
p	current directory
v	The version number
n	The default drive
g	The '>' character
l	The '<' character
b	The '||' character
_	A return and line feed
s	A space (leading only)
q	The '=' character
h	A backspace
e	ASCII code X'1B' (escape)

Examples

To change the prompt to display the current drive and directory and a greater-than symbol, enter

```
prompt $p$g
```

You can also create more complicated prompts. For example, to use the same prompt as above, with two added spaces and the characters "TIME =" followed by a display of the current time, enter

```
prompt $p$g  TIME=$t
```

The resulting prompt (if you are working in the "DOS" directory on drive C) will look like this:

```
C:\DOS>  TIME=11:07:54.23
```

RECOVER

Type: External

Syntax:

RECOVER [d:][path]filename

or

RECOVER d:

Purpose: Resolves sector problems on a file or a disk.

Discussion

If a disk develops problems, you can use this command to attempt to fix sector problems. You can recover either the file containing that sector (but without the bad sector) or the entire disk (if there were sector problems in the disk directory area).

Examples

To recover the file NEWFILES.TXT, enter

recover newfiles.txt

The program will read the file, sector by sector, skipping the bad sectors. When a bad sectors is found, the sectors are marked and DOS will no longer allocate your data to that sector.

To recover the disk in drive A, enter

recover a:

Each recovered file will be listed in the directory in the form FILEnnnn.REC, where nnnn is a sequential number starting with 0001.

If there is not enough room in the root directory, a message will be printed and the program will store information about the extra files in the file allocation table. You can run RECOVER later to regain these files (after you have removed files to make more room in the root directory).

REM [comment]

Type: Internal

Syntax:

REM [comment]

Purpose: Provides a way to insert remarks (that will not be acted on) into a batch file.

Discussion

During execution of a batch file, DOS will display (but not act on) comments which are entered on the line after the REM command.

You cannot use separators in the comment except the space, tab, and comma. To keep DOS from interpreting commands in a comment line, enclose the command in quotes.

Examples

To add a comment to a batch file, enter

```
REM  This is a comment line
```

To add a line to a batch file and to keep it from being displayed when the batch file is run, you must first turn the ECHO feature off. To keep the REM comment line from being displayed, enter

```
echo  off
rem  This comment line will not be displayed
```

RENAME (REN)

Type: Internal

Syntax:

RENAME (REN) [d:][path]filename filename

Purpose: Changes the filename under which a file is stored.

Discussion

RENAME changes the name of the first filename you enter to the second filename you enter. If you enter a path designation for the first filename, the renamed file will be stored on that same path.

Wildcard characters (* or ?) may be used with either the first or second filename designation.

Examples

To change the name of the file "SALES.TXT" on drive A to "NEWSALES.TXT" on drive A, enter

```
ren a:sales.txt newsales.txt
```

To change the filename extensions to .TXT on all files on drive B that have "SALES" as the main filename, enter

```
ren b:sales.* sales.txt
```

REPLACE

Type: External

Syntax:

REPLACE [d:][path]filename [d:][path][/A][/P][/R][/S][/W]

Purpose: Replaces stored files with files of the same name from a different storage location.

Discussion

This command replaces files in the target path designation with files of the same name in the source location. You can use wildcard characters (? and *) to indicate groups of filenames in the source directories.

You cannot use this command to replace hidden files or system files.

As files are replaced, the program displays their filenames on the screen. After the replace operation is complete, it displays

n file(s) added/replaced

After this command terminates, it sets one of the following exit codes:

0	Command completed successfully
1	Command line error
2	File not found
3	Path not found
5	Access denied
8	Insufficient memory available
15	Invalid drive

Options

/A Adds files specified by the source path designation that DO NOT exist on the target path designation. By using this option, you can selectively add files to a directory: a file in the source path designation that has the same name as a file on the target designation will not be affected.

This option cannot be used in combination with the /S option.

/P Displays a prompt before each file is replaced allowing you to decide on a file-by-file basis whether or not to allow the replacement.

/R Replaces both read-only and unprotected files on the target directory. Without this option, any attempt to REPLACE a read-only file will result in an error message and the replacement process will terminate.

/S Acts both on the target directory and all subdirectories within the target directory. You cannot use the /A option together with this option.

/W Waits for you to insert a disk before starting the replacement process.

Examples

To replace all copies of the file "MAIN.EXE" on all directories within the root directory of drive C with a file of the same name that is stored in the "SPECIAL" directory on drive A, enter

```
a: \special\main.exe  c:\/s
```

To add all files from the "NEWFILES" directory on drive A that have the .PCX filename extension to the "FILES" directory on drive C (as long as no file of the same name already exists there), enter

```
a:\newfiles\*.pcx  c:\files/a
```

RESTORE

Type: External

Syntax:

RESTORE d: [d:][path]filename[/P][/S][/B:mm-dd-yy]
[A/mm-d:d-yy][/E:hh:mm:ss][/L:hh:mm:ss][/M][/N]

Purpose: Restores to standard disk storage format files previously stored using the BACKUP command.

Discussion

When using this command, the first path designation should be for the BACKUP disk where the backed up files are stored; the second path designation should be for the target disk. This command transfers files (that have been previously backed up using the BACKUP command) from the backup disk to the specified target disk. Files backed up using the BACKUP command cannot be read (or otherwise used) until they are transferred from the BACKUP disk to the target disk.

If you do not specify a target path, files are placed in the current directory. If you do not specify filenames, backed up files from the specified path designation will be restored.

You can use wild card characters to indicate groups of files to be restored.

Options

/P The program prompts you before it restores files that have been changed since the last backup or files that are marked read-only. You can choose to continue the restoration of that file or not.

/S Backed up files from both the specified source directory and from sub-directories within that directory are transferred to the target path designation.

/B Only restores files that were modified on or BEFORE the date you enter.

/A Only restores files that were modified on or AFTER the date you enter.

/E Only restores files that were modified at or EARLIER than the time you enter.

/L Only restores files that were modified at or LATER than the given time.

/M Only restores files that have been modified since the last backup.

/N Only restores files that no longer exist on the destination disk.

ERRORLEVEL codes are set by the RESTORE command as follows:

0	Normal completion
1	No files found to restore
2	Some files not restored due to file sharing conflicts
3	Terminated by user (Ctrl Break or ESC)
4	Terminated due to error

Examples

To restore all the files from drive B to the root directory of drive C (and all sub-directories within it), enter

```
restore b: c:*.* /S
```

To restore the file "ANNUAL86" from the backup disk in drive A to the "FILES" directory on drive C, enter

```
restore a: c:\file\annual86
```

RMDIR (RD) (Remove Directory)

Type: Internal

Syntax:

RMDIR [d:]path

Purpose: Removes a subdirectory.

Discussion

RMDIR removes directories that are empty. To determine if a directory is empty, use the DIR command to display the directory's contents. For example, to determine if the directory "NEWFILES" is empty, enter

```
dir c:\newfiles
```

The display of an empty directory should look like this:

```
Volume in drive C is DOS1
Directory of C:\NEWFILES
.   <DIR>   4-09-87   8:36p
..  <DIR>   4-09-87   8:36p
2 File(s)   17671520 bytes free
```

For more information on using the RMDIR command and other fixed-disk commands, refer to Chapter 3, "Using Fixed Disks."

Example

To remove the directory "NEWFILES" from drive C, enter

```
rd  c:\newfiles
```

To remove the directory "SALES" that is within the "NEWFILES" directory on drive C, enter

```
rd c:\newfiles\sales
```

SELECT

Type: External

Syntax:

SELECT [d:] [d:][path][country code][keyboard code]

Purpose: Formats a disk and installs country-specific information and keyboard codes.

Discussion

SELECT transfers country-specific information, such as date and time formats for a country. After the target disk is formatted, the program transfers the CONFIG.SYS and AUTOEXEC.BAT files. Then the contents of the source disk are transferred to the target disk.

For more information on country codes, refer to Appendix B.

The source disk may be either drive A or drive B. If you do not enter disk designations, drive A will be assumed to be the source disk and drive B the target disk.

If you select a fixed disk as the target drive, the program will prompt you to type the correct label for that disk. If you enter the wrong label, the SELECT process is terminated. If you enter the correct label, the program will display this second warning:

```
WARNING, ALL DATA ON NON-REMOVABLE DISK
DRIVE D: WILL BE LOST!
Proceed with Format (Y/N)?
```

If you type N, the SELECT process will be canceled. If you type Y, the fixed disk will be formatted, erasing all the data currently stored on the disk.

Options

country code This is a three digit code to indicate the country information stored in the COUNTRY.SYS file (or a replacement file).

keyboard code This is a two character code to indicate the keyboard layout used. For more information, refer to the KEYB command.

Example

To create a new DOS disk in drive B with country-specific information for Spain, enter

```
select a: b: 850 sp
```

A warning will be displayed, giving you a chance to cancel the disk format before proceeding.

SET (Set Environment)

Type: Internal

Syntax:

SET
SET string1=string2

Purpose: Inserts strings into the command environment. The set values can be used later by programs.

Discussion

The SET command is used to set values that will be used by programs. DOS holds the set strings in the area of memory reserved for the environment (if the string already exists in the environment, it is replaced). After a string has been set in the environment, an application program can later access and use these strings. To use the second part of a set string (string2) the program will specify the first part of the set string (string1).

You can use spaces as part of set strings; therefore, do not enter a space before or after the equal sign (unless you intend to place a space in the string).

When the strings are placed in the environment, string1 is converted to uppercase. However, when referring to a set string you can use either uppercase or lowercase.

For more information on the SET command, refer to Chapter 5, "Tips for Advanced Users."

DOS also uses the environment internally to store data related to some previously used commands; for example, the path designation from a previously used PATH command is stored in the environment.

If you enter SET with no options, the current environment contents are displayed.

If you use the PROMPT or PATH commands, the parameters you set using these commands is also added to the environment (they too will be displayed when you enter SET with no options). Also, when DOS is started, it always stores in the environment a "COMSPEC" parameter. This describes the parameter DOS uses to reload the command processor (if necessary).

If you use terminate-and-stay resident (TSR) programs that "wait" in memory, a copy of the current environment is placed in memory.

If you enter the SET command with an existing string1 and an equal sign, the string is removed from the environment.

Example

You can use the SET command in batch files to define replaceable parameters by name instead of by the number of a passed parameter. You can, for example, place a variable name such as %file1% in your batch file (note the percent signs before and after). Then, if you enter the line

```
set file1=newfile.doc
```

the filename "NEWFILE.DOC will be used as the %FILE1% variable when you run your batch file. You can easily change the %FILE1% variable in the batch file as often as needed by using the SET command.

You can also use write executable programs that look to the environment for variables. For example, a variable used by a printer control program to set the printing width (page size) might be set in the environment as

```
set width=wide
```

SHARE

Type: External

Syntax:

SHARE [/F:(space)] [/L:(locks)]

Purpose: Installs support for file sharing and file locking.

Discussion

The SHARE command is used only when networking is active. After you use this command, DOS will check all read or write requests to determine the file sharing or locking status.

The SHARE command can be included in the AUTOEXEC.BAT file to install shared files.

Options

/F:space Allocates file space (in bytes) for the area DOS will use to record file sharing information. Each file that is opened needs the length of the full filename plus eleven extra bytes of memory (the average is 20 bytes). If you do not specify a value, it is set to 2048 bytes.

/L:locks Allocates the number of locks to allow. If you do not specify a value, it is set to 20 locks.

Example

To load file sharing and use the default values for the /F and /L switches, enter

share

SHIFT

Type: Internal

Syntax:

SHIFT

Purpose: Increases number of replaceable parameters to more than the standard ten for use in batch files.

Discussion

Usually, you can only enter up to ten parameters at the time you start a batch file (%0 through %9). Use this command to shift the assignable parameters to make room for the use of additional entries.

Example

If you started a batch file named "GO.BAT" by entering the following on the command line:

go X1 X2 X3 X4 X5 X6 X7 X8 X9 X10

the parameters available for use in the batch file "GO" will be set as follows:

%1 = X1
%2 = X2
%3 = X3
%4 = X4
%5 = X5
%6 = X6
%7 = X7
%8 = X8
%9 = X9

To use the additional parameter you entered on the command line (10), enter

shift

The parameters will be reset as

%1 = X2
%2 = X3
%3 = X4
%4 = X5
%5 = X6
%6 = X7
%7 = X8
%8 = X9
%9 = X10

The set parameters are shifted one higher, giving you access to more than ten parameters. Each time you enter the SHIFT command, the parameters used by the batch file is shifted up one position.

SORT

Type: External

Syntax:

SORT [/R][/+n] < (name)

Purpose: Sorts input and sends it to the screen or to a file.

Discussion

SORT is a "filter" command (reads from input, transforms it, and outputs it to the screen, to a file, or to a printer). SORT is used to alphabetize a file. You can specify which column in the file to sort on. If you do not specify a column, SORT alphabetizes using the character in the first column.

If you do not specify a source file, SORT can be used with input from the keyboard or another input device.

Use the | and < redirection symbols to direct the sort output to a new file. For more information, refer to Chapter 5, "Tips for Advanced Users"

SORT does not alphabetize upper case letters differently from lower case letters.

Special characters (above ASCII value 127) are sorted according to character information found in the COUNTRY.SYS file (or in a replacement file as designated in the CONFIG.SYS file). For more information, refer to Appendix B, "Country-Specific Information."

Options

/R Reverses the sort (sorts from Z to A).

/+n Starts sorting with column number (n) you enter.

Examples

To sort the file "NAMES" on drive A and send the output to a new file "NAME2" on drive B, enter

```
sort < a:names > b:name2
```

To reverse sort (Z to A) the file "PHONES.TXT" on drive A and send the output to a new file "PHONE-A" on drive C, enter

```
sort /r < a:phones.txt > b:phone-a
```

To create a sorted file "TEMP," sorting on the character in column three, by using input from the keyboard, enter

```
sort /+3 > b:temp
```

Then, enter lines and press the **Enter** key at the end of each line. When you are finished, press and hold the **Ctrl** key and then press the letter **Z** key. The data you enter will be sorted (based on the third character you enter on each line) and the "TEMP" file will be created and stored on the current drive.

You can also "pipe" the output of a SORT to a file. In the following example, the directory listing will be sorted starting with column 14 (the column in the directory list that contains the file size). The output will be sent to the file "DIR-S" on the current directory.

```
dir | sort /+14 > dir-s
```

SUBST

Type: External

Syntax:

SUBST d: d:path
SUBST d:/D

Purpose: Substitutes a virtual drive letter for a path designation.

Discussion

Use the SUBST command to substitute a drive letter for a path in order to treat a virtual drive (a reserved area rather than an actual disk drive) as a physical drive.

In order to enter drive designations using any letter (other than just the letters for the physical drives installed in your computer), you may have to add this line to a CONFIG.SYS file that should be placed in the root directory of your startup drive:

LASTDRIVE=Z

Six DOS commands cannot be used with virtual drives created with the SUBST command. They are:

CHKDSK
DISKCOPY
FDISK
FORMAT
LABEL
SYS

If you enter the SUBST command without options, the program will display the name of the current virtual drives that are in effect (if any).

For more information on using the SUBST command and other fixed-disk commands, refer to Chapter 3, "Using Fixed Disks."

Example

To create a virtual drive F for the path, "SALES\LETTERS" (under the root directory) of drive C, enter

subst f: c:\sales\letters

Read and write requests to either drive F or to C:\SALES\LETTERS will be stored in the C:\SALES\LETTERS directory area.

SYS (System)

Type: External

Syntax:

SYS d:

Purpose: Transfers the operating system files to another disk.

Discussion

You can use the SYS command to transfer system files to a disk only if:

- The disk has been formatted but DOES NOT contain files or a volume label.
- The disk has been formatted using either a /S or a /B format option.

You cannot reliably transfer system files to a disk that already has a different version of DOS on it.

The system files are hidden files that do not appear when you use the DIR command to display a directory of a system disk.

When you use the SYS command to transfer the system files to a disk, the file COMMAND.COM (which contains the command processor) is not transferred. You must use the COPY command to transfer COMMAND.COM.

A system disk can also be transferred (along with all the files stored on it) using the DISKCOPY command.

If an error is encountered, SYS will not transfer the system files. DOS will report one of the following error messages:

Message: No room for system on target disk

Meaning: This message means that there is not enough room on the target disk for the system files. It does not necessarily mean that the disk is already filled with stored files; it may mean that the disk was not formatted using the /S or the /B format options (both format the disk and reserve space for the system files).

Message: Incompatible system size

Meaning: The system files do not take up the same amount of space on the target disk as the new system will need. This can happen if you try to transfer system files to a disk that already has a different version of DOS on it.

TIME

Type: Internal

Syntax:

TIME

TIME hh:mm[:ss][.cc]

Purpose: Displays current time setting of system clock and provides a way for you to reset the time.

Discussion

You can enter the TIME command alone and the program will prompt you when to enter the time. You can also enter the time at the same time you enter the time command. Either way, you must enter the time in the following format:

hh:mm:ss.cc

where hh is a one or two digit hours designation, mm is a one or two digit designation of minutes after the hour, ss is a one or two digit designation of seconds after the minute, and cc is a one or two digit designation of the hundredths of a second past the second. The following numbers are allowed:

hh	0-23 for hours
mm	0-59 for minutes
ss	0-59 for seconds
cc	0-99 for hundredths of seconds

Only the hour setting and the minute setting are required. The following are allowable settings:

04:23
4:03
11:5:09
17:59:02.22
2:3:2.9

> **NOTE:**
> Your version of DOS may require slightly different forms of this entry format.

If you enter the TIME command without any time designation, the screen will display:

Current time is hh:mm:ss
Enter new time:__

You can enter a new time or press the **Enter** key to leave the time setting as it is.

If DOS carries out a time-dependent function (such as time-stamping a file for storage on a disk), the system time (based on the starting time you set) will be used. On some computers the system time is maintained by a battery-backup system; in this case, you will rarely use the TIME command (you can stop the initial presentation of both the DATE and TIME prompts by creating any AUTOEXEC.BAT file that does not include these commands). Other computers "lose" the system time whenever the computer is turned off. In this case, it must be reset each time the computer is restarted.

TREE (Display Directory)

Type: External

Syntax:

TREE [d:][/F]

Purpose: Displays directory paths and (optionally) files in each subdirectory.

Discussion

When you use the TREE command each directory name is displayed along with the name of any subdirectories within it. The display will be in a format like the summary below. (Different versions of DOS may display the data in a slightly different format.)

First, the root directory and the directories within it are listed (directory names listed in a five column display):

DIRECTORY PATH LISTING FOR VOLUME (volume name)

Path is C:

Then each directory within the root is listed:

Path is: C:\(directory name)

Option

/F Displays the names of the files found within each directory listed.

For more information on using the TREE command and other fixed- disk commands, refer to Chapter 3, "Using Fixed Disks."

Example

To list the directories on the disk in drive B, enter

tree a:

TYPE

Type: Internal

Syntax:

TYPE [d:][path]filename

Purpose: Displays the contents of a file.

Discussion

When you use the TYPE command, the file is displayed with limited on-screen formatting. Tabs are expanded and generally displayed as eight spaces wide. If you display files that contain special (non-text) characters, these characters may have unpredictable effects on your display.

Wild card characters (? and *) cannot be used with this command in either the filename or the extension.

Example

To display the contents of the file LETTER3.TXT on drive B, enter

type b:letter3.txt

VER

Type: Internal

Syntax:

VER

Purpose: Displays the DOS version number.

Discussion

VER displays the DOS version number for the version of DOS currently active.

Example

To display the currently operating version of DOS, enter

ver

The program will display

MS-DOS Version (version number)

VERIFY

Type: Internal

Syntax:

VERIFY

VERIFY ON ¦ OFF

Purpose: Verifies that data written to disk was correctly recorded.

Discussion

The VERIFY command has the same purpose as the /V parameter in the COPY command; once set, you do not have to use the /V option while copying. After VERIFY has been set to ON, DOS checks for bad sectors following all data writes to disk. This can cause a slowing of such operations.

An error message is displayed only if DOS was unable to successfully write your data to disk.

If you enter the command with no options, DOS will display the current VERIFY setting.

Example

To tell DOS to verify all writes to disk, enter

```
verify on
```

DOS will check to be sure that each write to disk was carried out successfully. This condition remains in effect until you enter

```
verify off
```

VOL (Volume)

Type: Internal

Syntax:

VOL [d:]

Purpose: Displays a disk's volume label.

Discussion

The VOL command is used to display the volume label. It is not used to add or change the volume label on a disk. For information on how to create, change, or delete the volume label, refer to the LABEL command in this chapter.

Example

To display the volume label for the disk in drive A, enter

vol a:

the program displays:

Volume in drive A is (label)

If the designated disk does not have a volume label, the program displays:

Volume in drive A has no label

XCOPY

Type: External

Syntax:

XCOPY [d:][path]filename [d]:[path][filename]
[/A][/D:(date)][/E][/M][/P][/S][/V][/W]

Purpose: Copies directories, subdirectories, and files.

Discussion

XCOPY copies files or groups of files to and from directories. Especially useful for copying data to and from a fixed disk.

The first path designation specifies the source file(s); the second path designation specifies the target file(s). A source designation must always be provided. If you do not enter a target designation, files will be copied to the current directory.

This command provides a way to copy entire disks full of data, including all established directories and subdirectories. This command should be used instead of the DISKCOPY command when you want to create a disk with identical file structures, but with a different disk format (for example, to copy all files from a 5 1/4 inch disk to a 3 1/2 inch disk).

If the source is a directory, the directory will be copied onto the destination diskette. If the source to be copied includes more than one file, the destination files will be stored in a directory. You can add a backslash (\) to the end of the destination filename to make it a directory.

If XCOPY does not find a directory on the destination disk that matches a PATH specification in the command format, the copying will stop. A message will be displayed asking you to specify whether or not the destination file specification is to be stored as a filename or a directory name.

XCOPY is a much more powerful copying command than the COPY command. It is especially useful when backing up your fixed disk. A common usage would be to use the "/S" option along with the "/D:date" option to backup only the files that have been changed since the date of your last backup. If you are using XCOPY for this purpose, you will probably want to make a copy of the program (XCOPY.EXE) in the root directory of your fixed disk (remember, when you use the /S option, XCOPY only copies files from subdirectories found WITHIN the current directory.)

After you enter the XCOPY command, the program may display

Does TST specify a file name
or directory name on the target
(F = file, D = directory)

This means XCOPY cannot determine if you want the specified path designation entry to be sent to a filename on the target disk or to a directory.

NOTE:

Some versions of DOS provide a way to avoid this prompt. It is done by renaming the XCOPY.EXE command to MCOPY.EXE. When using MCOPY, DOS will automatically check to determine if the target is a filename or a directory name and carry out the copy accordingly. To change the name of the command to MCOPY, enter

copy xcopy.exe mcopy.exe

Then, you can use the new "MCOPY.EXE" program just as you would use "XCOPY.EXE." The only difference is that MCOPY will automatically determine whether the target is to be a file or a directory.

If an error is encountered during the XCOPY process, the program returns one or the following exit codes:

0 Copying was completed without error
1 No files found
2 User terminated the copy
4 Initialization error (not enough memory, invalid syntax, path not found)
5 Copy terminated due to INT 24 error reading or writing disk

For more information of using the XCOPY command, refer to Chapter 5, "Tips for Advanced Users."

Options

/A Copies only files that have been set as archive files (using the ATTRIB command). The copied files will still be marked as archive files in BOTH the source and destination files.

/D:(date) Copies only those files in the source directory that have been changed on or LATER than the specified date.

/S Copies all files in the current directory and in any subdirectory within it.

/E Copies subdirectories, even if they are empty. If you use this option, you must use the /S option also.

/M Copies the same marked archive files as specified in the /A option; however, the files on the destination disk are no longer marked as archive files. This switch checks the archive attribute of a file: if the file's archive attribute is set to off (-A), the file will not be copied.

/P Does not copy the file until you respond to a prompt that asks you to confirm that you want the file copied. However, if you are using the /E option, you will not be prompted before an empty subdirectory is copied.

/V Each file copied is verified to be sure the file stored on the destination disk is identical to the original on the source disk.

/W Copies only after you respond to a prompt to begin copying (if this option is omitted, the copying begins as soon as you enter the XCOPY command and press the **Enter** key).

Example

To copy all files and directories (except empty directories) from drive C to drive A, enter

xcopy c: a: /s

Appendix A
DOS Error Messages

A number of common errors can occur while using DOS commands. Unfortunately, the error messages that DOS displays are not always easy to understand. A few of the more common error messages (and what to do about them) are described in the following sections.

The error messages described in this appendix are standard for most implementations of MS-DOS. However, different manufacturers may rewrite the messages to be more specifically relevant to their equipment. For this reason, the exact wording of the error message shown may be somewhat different from the messages displayed by your computer system.

Common Error Messages

Message: Abort, Retry, Ignore?

This is one of the most common (and yet one of the most worrisome) of all the DOS error messages. It often accompanies another error message that indicates a specific type of error.

What to Do About It

One common problem that produces this error message is when DOS cannot read correctly from a drive while trying to carry out a DOS command. In this case, check to be sure the disk in the drive has been formatted properly. Try again with another diskette that you are sure has been formatted for DOS and for your type of computer. If possible, try the troublesome disk in another computer.

Check the drive to be sure the drive door is closed securely. Make sure the drive door latch is locked.

Check to be sure that the disk was inserted properly. If you are using 5 and 1/4 inch flexible diskettes (floppies), the diskette can be inserted incorrectly; the correct way

is with the label up, the oblong window in the plastic jacket inserted first, with the notch (if the disk has one) to the left.

If you have more than one diskette drive, try again and watch to see which light on the front of the drive comes on.

Unfortunately, this error message can also occur when there is a problem with the data stored on the disk. The message is sometimes combined with an error message similar to the following:

Message: **Disk error while reading (or writing) drive A**

As indicated by this message, the problem lies in how the data is stored on the disk. Either DOS cannot find all of its proper storage location indicators or there is a physical problem with the disk.

You may be able to make a copy of the disk and try again, but this rarely works. If you are using a diskette, remove it and look for physical damage. If the disk is damaged, do not attempt to use it again. If the disk is dirty, do not use it any longer. (Unfortunately, there is no effective way to clean a diskette).

There are some utility programs on the market that will allow you to "go around" DOS and look at the data on the disk sector by sector. By using one of these programs, you may be able to retrieve some of the data on the disk.

This type of error message can also occur if DOS had trouble communicating with a peripheral device such as a printer. Make sure the device is turned on. If it is a printer, make sure it is "on line" and has paper loaded.

Check to be sure the power plug is plugged in. Check to be sure all data cables are connected properly and that they are of the correct type.

After you have corrected the problem (or are satisfied that you can't correct it), you have -- as the error message indicates -- three options. They are:

Abort This option allows you to cancel the entire operation. If this is what you want to do, press the letter A key. DOS will no longer attempt to carry out the DOS command.

Retry This option tells DOS to again attempt to carry out the failed operation. DOS does not attempt any error checking or alternate means of recovery from the error; it merely tries again to carry out the previously specified DOS command.

Ignore This option tells DOS to ignore the error condition and continue with the interrupted DOS command. Of the three options, this is the most dangerous. If, for example, DOS was attempting to read from the disk or write data to the disk, it will continue with the attempt even if the proper storage location indicators are missing. This will usually result in lost or scrambled data. Use this option only as a last resort.

Message: Invalid file name

This error message indicates that DOS was not able to carry out the specified command because the named file could not be found.

What to Do About It

You may have used illegal characters in the filename specified or used a reserved filename. Refer to the "Filenames" sections in Chapter 1 for more information about illegal and reserved filenames.

If you were asking DOS to take an action on an existing file, you may have misspelled the filename. Check the spelling character by character, including the three character extension to the filename.

You may not have given DOS enough file specification information to find the file named. Be sure to include full path information with properly specified directory and subdirectory information. Refer to Chapter 2 for more information on path names.

Message: Disk is write protected

As part of DOS a operation (or a DOS-accessed operation by another program), an attempt was made to write data to a diskette that is write-protected.

What to Do About It

Disks are write protected in one of three ways:

1. The square notch cut out at the upper right of a 5 and 1/4 inch diskette is covered by tape or by a sticky-backed tab. This write-protect tab must be removed before DOS can write data to the disk. Many disks are write-protected to protect irreplaceable data that is stored on the disk. If you are not sure you want to risk loosing the data stored on the disk, it is best to make a copy of the disk (using either the COPY command or the DISKCOPY command) and then use the copy instead of the original.

2. An original program disk may not have a square write-protect notch cut out of the upper right area of the diskette. In this case, the disk is permanently write-protected and data cannot be written to the disk. You must make a copy of the disk (using either the COPY command or the DISKCOPY command) and then use the copy instead of the original.

3. The smaller 3 and 1/2 inch, hard-shelled diskettes do not have a cut out notch for write protection. Instead, there is a square hole in the disk case that can be covered by sliding a plastic cover in place over it. The disk is write-protected by sliding the cover toward the outside edge of the disk.

Message: **Not ready error**

This error message indicates that a DOS command was not carried out because DOS received a "not ready" indication from a drive or from another device.

What to Do About It

The most common cause of this error message is an attempt to carry out a DOS command on a disk in a drive that is not ready to receive information from the computer. This can mean that the diskette has not been inserted properly or that the drive is not closed and latched correctly.

Refer also to the error message "Abort, Retry, Ignore" described above.

Message: **Seek error**
or
Sector not found

These error messages indicate that a DOS command was not carried out because DOS could not locate the proper track on the disk.

What to Do About It

The problem is that DOS cannot locate the proper disk storage information when trying to access the disk. The storage location information that DOS always stores on a disk to keep track of the locations of all files could not be accessed.

It could mean that the disk has been damaged. Check the diskette for signs of dirt or damage. If the diskette is dirty or if it has been damaged, do not attempt to use it again.

The message can also mean that all or part of the data on the disk was not stored properly in the first place. Something could have happened to cause DOS to write the data in a disk location than could not be relocated later. Try reading other files on the disk. If other files can be read, copy the troublesome file to a new disk and try the operation again.

This type of error can happen for many reasons. If it happens often, you may have a problem with your disk drive or a problem with the type of diskette you are using. Make sure you are using the correct density diskettes for the type of diskette drive used by your computer. Make sure the diskettes were formatted originally for the type of diskette drives your computer uses.

Refer also to the error message "Abort, Retry, Ignore" described above.

Message: **Bad command or filename**

This message usually means that DOS tried to load and execute a program file. The specified filename could not be found.

What to Do About It

This is one of the most common DOS error messages, usually because you have made a mistake. It happens most often when you either misspell the name of the program you wanted to run or because you did not give DOS enough information to be able to find the program.

When you enter any series of numbers or letters that DOS does not recognize as one of its own command names, DOS automatically assumes that you are trying to load and run some other type of program. DOS will then look for a program name as indicated by the set of characters you entered. Specifically, it will look for a program by that name with either an .EXE extension, a .COM extension, or a .BAT extension. DOS will first look for a program by that name, with one of the three extensions to the filename, stored on the current disk in the current directory or subdirectory. If it does not find a program by that name, it will also search the root directory and any other subdirectories that have been previously specified by the PATH command. If all of these searches fail, DOS issues the "Bad command or filename" message.

Message: **Bad or missing Command Interpreter**

This message can occur when the computer is started or reset. It means that DOS tried to load its own system files and could not find them in the default drive.

What to Do About It

In order to start a computer that uses the MS-DOS operating system, you must have a "bootable" disk in drive A or else be using a fixed disk that has been formatted as a MS-DOS system disk.

When the computer is first started (or reset), DOS will first check to see if a diskette is in drive A. If it is, DOS will look for its own system files on the diskette. If the system files are found there, they will be loaded into the computer's memory and the A prompt will be displayed to indicate that the computer is ready to receive commands. If a diskette is found in drive A but the system files are not found on the disk, DOS will issue this error message.

If a diskette is not found in drive A, DOS will next check to see if a fixed disk is installed in the computer. If it is found, DOS will search the root directory of that drive for its system files. If system files are not found on the fixed disk, DOS will issue the error message.

This message can also occur if you removed the DOS system disk from the drive or directory it was in when DOS was started. Some DOS activity can occasionally require DOS to re-access its own system files. If, during this process, DOS cannot find these files, this message will be displayed.

Make sure the DOS system files are available either on the disk in drive A or on the root directory of the fixed disk, or have it available when DOS asks for it.

If a copy of the DOS system files are available to DOS and this message is still displayed, there may be a problem with the system files themselves. Use the original DOS disk to recreate the system files on the disk. It's done by entering

sys d:PATH

The "d:" refers to the drive you want the system files transferred to. PATH refers to the subdirectory.

This error can also be caused if the COMSPEC command directs DOS to the wrong directory.

Message: **File cannot be copied onto itself**

This is a common error message that is displayed when DOS attempts to carry out a COPY function, but no location for the copy was specified other than the current disk or directory.

What to Do About It

This error message is commonly the result of forgetting to specify where the result of a COPY command should be stored. For example, if you enter

copy test1.txt

you have not specified where the copy of TEST1.TXT should be stored. There would be no point in copying the exact duplicate of the file on top of itself (in the same location on the disk) therefore, DOS takes no action and displays the error message.

When copying, you have two options: you can either

1. Change the name and make a duplicate copy on the same diskette or directory under the new name.

2. Copy the file to a different disk or a different directory with the same name.

Message: File not found

This message is displayed when DOS attempts to carry out an action on a file (or files) you have named.

What to Do About It

This is a very common message that usually does not refer to an error condition; most likely, the file you named does not exist on the disk.

Check to see if the filename you specified was spelled correctly. Make sure you have provided DOS with the correct pathnames.

DOS may also display this message as a proper response to a command you typed in. For example, if you entered

dir b:test*.*

you are asking DOS to display all files that begin with the letters "TEST" on the disk in drive B. If DOS does not find any files that start with those particular letters, it will display the "File not found" message. As you can see, it is not reporting an error, but is accurately reporting the result of the search.

Message: Insufficient disk space

This is a common error message that is displayed when DOS attempts to carry out a COPY function, but finds that there is not enough space on the destination disk for the copied file.

What to Do About It

DOS will copy files onto a disk until it runs out of room. If you specified more than one file to be copied, some files will be transferred. However, no part of the file that DOS was attempting to copy and store on disk will be transferred. You will either have to clear out some space on the disk by using the ERASE command or else you will have to provide another disk.

Message: Out of paper

This refers to a printer error. It means that the printer has a switch that senses when there is not paper in the paper feed track and this switch is sending an error condition back to the computer. (This situation can also produce other types of error messages that do not mention the paper problem.)

What to Do About It

Check to be sure paper is in the feed track. The paper-sensitive switch may be under the paper rollers so be sure the paper in the printer is long enough to reach the switch. If there seems to be plenty of paper, the switch may be jammed. Try turning off the printer and rolling some paper through the printer by hand.

Once the paper sensitive switch has been activated, it may need to be "cleared." You may have to turn the printer off and on again (and/or the computer) to reset it.

Message: Syntax error

This is a common error message that tells you that DOS could not act on the command you specified because the DOS command was not entered correctly.

What to Do About It

If you get this message, it means that DOS WAS able to determine which command you entered, but there is a problem in the way you entered the command..

Re-enter the command, paying special attention to the exact command syntax. Also check the command syntax for each option you are using with the command. Remember, each space (or non-space), as well as other special characters have

meaning when used in DOS commands. Refer to Chapters 1 and 3 for more information on entering DOS commands.

Problem Solving

Many of the "fixes" described in this appendix may also be effective for other types of errors. When trying to track down the cause of an error message, proceed systematically. Look for obvious causes before deciding on more complicated reasons. Write down the exact text of the error message and analyze it. If possible, try to make the error happen again.

As we pointed out in chapter three, many DOS errors have something to do with disks or disk drives. Always check to see that a diskette has been inserted properly. Make sure the diskette is of the correct type for your computer and be sure that it has been properly formatted for your computer drives. Try another diskette. Whenever possible, try the diskette in another computer. In fact, if possible, try repeating the entire problem task on another computer.

The basic role of a trouble-shooter is to "try something else." In fact, you should try everything you can think of. It is all but impossible to damage a computer with anything you can type in from the keyboard; in fact, protecting the computer is one of DOS's main tasks. It stands between you and the basic functions of the computer. The error messages are one indication of this protective function.

You can save yourself time and money (and learn a lot about DOS in the process) by being your own trouble-shooter. Try to eliminate every possible combination of possibilities before calling for help. You'll be amazed at how much you can learn on your own.

Additional Error Messages

In addition to the list of common error messages discussed in the previous section, DOS may issue many other messages. Many of them are difficult to interpret, but, fortunately, they usually do not occur very often. This section provides an alphabetical listing of these messages along with a brief explanation.

Message: **All specified files are contiguous**

Explanation: You used the CHKDSK command with one or more filenames. It indicates that all files are allocated contiguously on the disk without the need for fragmentation.

Message: **Allocation error in file, size adjusted**

Explanation: The size of the space reserved for a file block is incorrect. DOS determined there was an error condition because the size of the file indicated in the directory was not consistent with the amount of data actually allocated to the file. If you run CHKDSK using the /F option, it will alter the size allocation number.

Message: **All specified files are contiguous**

Explanation: You used the CHKDSK command with one or more filenames. It indicates that all files are allocated contiguously on the disk without the need for fragmentation.

Message: **Bad call format reading drive**

Explanation: An error occurred while DOS was reading from a device.

Message: **Bad call format writing drive**

Explanation: An error occurred while DOS was writing to a device.

Message: **Bad file**

Explanation: A file you specified while using a DOS command is defective.

Message: **Bad or missing filename**

Explanation: DOS found an invalid device named in the CONFIG.SYS file. Check the "DEVICE = " line in the CONFIG.SYS file.

Message: **Bad or missing Command Interpreter**

Explanation: DOS cannot find the COMMAND.COM file in the current path (usually the root directory of the disk being used). This can mean that the file is missing, or the file is not the version currently in use. Usually, you will have to restart the system and then copy a valid version of the COMMAND.COM file to the root directory of the disk used to start DOS.

Message: **Bad unit error reading drive (d:)**

Explanation: An error occurred while DOS was reading from a device.

Message: **BREAK is ON or OFF**

Explanation: Reports the current setting of BREAK.

Message: **Cannot CHDIR to filename tree past this point not processed**

Explanation: The CHKDSK command was in operation, but DOS was unable to verify the directories below the directories specified to this point.

Message: **Cannot CHDIR to root**

Explanation: The CHKDSK command was in operation, but DOS was unable to process the directories below the directories specified to this point. The CHKDSK command terminates after this message is issued.

Message: **Cannot CHKDSK a Network drive**

Explanation: You cannot check drives which have been redirected over a network.

Message: **Cannot do binary reads from a device**

Explanation: This message appears when you are using the COPY command with the /B option. The COPY cannot be completed in binary mode when you are copying from a device. Remove the /B switch or copy the file using the /A option.

Message: **Cannot format an ASSIGNed or SUBSTed drive**

Explanation: You can't use the FORMAT command on a drive that has been mapped to another drive with the ASSIGN or the SUBST command.

Message: **Cannot FORMAT a Network drive**

Explanation: You cannot format drives that have been redirected over a network.

Message: **Cannot open (filename)**

Explanation: When using the PRINT command, either DOS can't find the specified file to print or the file does not exist.

Message: **Cannot recover . entry, processing continued**

Explanation: The CHKDSK command encountered an error when attempting to check the specified directory. Nevertheless, the CHKDSK command will continue to check the disk.

Message: **Cannot Recover a Network drive**

Explanation: You cannot recover files on drives that have been redirected over a network.

Message: **CHDIR failed, trying alternate method**

Explanation: During the operation of the CHKDSK command, an error was encountered when the search process attempted to return to a parent directory. The command will continue by attempting to return to that directory by starting over at the root directory and traveling down.

Message: **Content of destination lost before copy**

Explanation: A file that was specified as a source file to the COPY command has been overwritten prior to completion of the copy. You can't use a destination filename that is the same as one of the source filenames. For example, if you enter

```
copy file1+file2 file2
```

DOS will attempt to combine "FILE1" and "FILE2" into a new file. However, the source filename "FILE2" will be overwritten and lost before it can complete the copy.

Message: **Convert lost chains to files (Y/N)?**

Explanation: The CHKDSK command has encountered lost chains while checking the disk. Press the letter **Y** key to create new files and store the lost segments in them. The names of the new files will start with the letters "FILE" and end with a four digit number. Press the **N** key to tell DOS NOT to save the lost segments. DOS will free the allocated blocks so that the disk space occupied by them can be overwritten by other files.

Message: **Corrections will not be written to disk**

Explanation: The CHKDSK command found errors on the specified disk, but the /F switch was not specified; therefore, CHKDSK will not write lost segments to new files.

Message: **Data error reading drive (d:)**

Explanation: An error occurred while DOS was reading from a device.

Message: **Directory is joined**

Explanation: The CHKDSK command attempted to act on a path designation that includes a drive that has been specified with the JOIN command: CHKDSK will not work on directories that have been JOINed.

Message: **Directory not empty**

Explanation:

The JOIN command attempted to join a directory which was not empty. This is not allowed.

Message: **Directory is totally empty, no. or..**

Explanation: The CHKDSK command has found that the specified directory does not contain the required references to a working and parent directory. You must use RMDIR to remove the directory and then recreate it.

Message: **Disk error reading drive (d:)**

Explanation: An error occurred while DOS was reading from a device.

Message: **Disk error reading FAT (d:)**

Explanation: The CHKDSK command found an error while checking the file allocation table (FAT). CHKDSK will continue to operate using the backup FAT (if available). To be safe, you should copy the files from the specified disk to another disk.

Message: **Disk error writing drive (d:)**

Explanation: An error occurred while DOS was reading from a device.

Message: **Disk error writing FAT (d:)**

Explanation: The CHKDSK command found an error while checking the file allocation table (FAT). CHKDSK will continue to operate using the backup FAT (if available). To be safe, you should copy the files from the specified disk to another disk.

Message: **Disk unsuitable for system drive**

Explanation: The FORMAT program detected a bad track on the disk where it was attempting to write the system files. However, the disk can still be used to store data files.

Message: **Disks must be the same size**

Explanation: The DISKCOPY command has attempted to copy to a destination disk that is of a different type than the source disk. This is not allowed. However, you can still copy the files from the source disk to the destination disk using the COPY command.

Message: **Divide error**

Explanation: DOS has encountered a divide overflow error. This is caused by an internal mathematical processing problem, such as an attempt to divide by zero.

Message: **Divide overflow**

Explanation: DOS has encountered a error which sets the divide overflow flag (a system error notification that stops processing). This is caused by an internal mathematical processing problem, such as an attempt to divide by zero.

Message: **Do not specify filename(s)**
Command format: DISKCOMP d: d: [/1][/8]

Explanation: The DISKCOMP command has determined that you entered an incorrect option. It generally means that you attempted to enter a filename as well as a drive name. This is not allowed.

Message: **Does command (name) specify a file name or directory name on the target (F = file D = directory)?**

Explanation: The XCOPY command has determined that a directory with the specified name does not exist on the target disk. Therefore, DOS cannot determine whether or not the name you specified is to be created as a directory or as a file. Enter **F** to instruct DOS to create a file by the specified name, or **D** to create a directory by that name.

Message: **(.) (..) Does not exist**

Explanation: The CHKDSK command is reporting invalid entries for the required working and parent directories. However, CHKDSK will continue.

Message: **Duplicate file name or File not found**

Explanation: The RENAME command attempted to rename a file to a filename you specified; however, the filename either already exists on the target directory or the name you specified for the source file could not be found.

Message: **Entry has a bad attribute**

Explanation: There are a number of possible errors having to do with DOS's internal record of a file's attributes. For example, a size of linking attribute associated with one or more files does not match the actual file information.

If you run CHKDSK using the /F option, it will try to correct the error. If a directory is involved, this message may be preceded by one or two periods (to indicate which directory is invalid).

Message: **Error writing to device**

Explanation: DOS was unable to write to the device you specified.

Message: **Errors found, F parameter not specified**

Explanation: The CHKDSK command found errors, but will not carry out corrections unless you run the command again specifying the /F option.

Message: **Errors on list device indicate that it may be off-line**

Explanation: DOS is not able to detect a response from your printer. It may be turned off or the on-line switch may be set to off.

Message: **File allocation table bad**

Explanation: The file allocation table on the specified drive has been found to be defective. Running CHKDSK may solve the problem.

Message: **File creation error**

Explanation: DOS has failed to create the specified file. This may mean that you created a file name that already exists in the directory. If this is not the problem running CHKDSK will help you determine the cause of the error.

Message: **First cluster number is invalid, entry truncated**

Explanation: DOS displays this message when you use the CHKDSK command and DOS's internal directory of file information on the disk is found to contain a pointer to file storage information that does not match the actual files stored on the disk. If you run CHKDSK using the /F option, the pointer information is corrected (truncated to a zero length file).

Message: **For cannot be nested**

Explanation: You have created a version of the FOR commands that is within another FOR command (referred to as "nested"). This is not allowed.

Message: **Format failure**

Explanation: The FORMAT command failed to completely format the disk. The FORMAT command may also display other explanations as to why the FORMAT failed.

Message: **General failure reading drive (d:)**

Explanation: DOS was unable to read any of the file storage information on the specified disk. This message usually indicates that DOS will not be able to access any of the data on the disk. Running CHKDSK will probably NOT resolve this type of disk error.

Message: **General failure writing drive (d:)**

Explanation: DOS was unable to write to the specified disk. This message usually indicates that DOS will not be able to access any of the data on the disk. Running CHKDSK will probably NOT resolve this type of disk error.

Message: **Incorrect DOS version**

Explanation: Many of the commands included in the newer versions of DOS will not run if you have loaded on older versions of DOS. For example, the DOS commands, CHKDSK, PRINT, and SYS will only run under the exact version of DOS for which they were configured. However, you can load and run all of the newer versions of DOS on most older IBM-compatible computers

Message: **Incorrect number of parameters**

Explanation: You specified too many or too few options as part of a DOS command syntax.

Message: **Insert diskette with batch file and press any key when ready**

Explanation: The disk containing the batch file is no longer in the drive currently accessed. Make sure the disk containing the batch file is still in the drive it was in when you started the batch process.

Message: **Insufficient memory**

Explanation: There is not enough system memory (RAM) to perform the specified operation.

Message: **Insufficient memory for system transfer**

Explanation: There is not enough system memory (RAM) to perform a system transfer.

Message: **Insufficient room in root directory**

Explanation: The CHKDSK command has failed to write new files to the root directory because of insufficient storage space. CHKDSK attempts to recover lost files by creating new ones in the root directory. You must erase some of the files in the root directory and rerun the CHKDSK command.

Message: **Invalid characters in volume label**

Explanation: The FORMAT command has failed to create a volume name because of a problem with the label you specified. The volume label should contain only up to 11 characters and should not use any of the reserved characters (refer to Chapter 1 for a list of the allowable characters).

Message: **Invalid COMMAND.COM**

Explanation: DOS has attempted to access the memory resident portion of the COMMAND.COM program. This often occurs when the program you are using used up too much memory. You must insert the disk with the COMMAND.COM program so DOS can reload system components from the command file. If processing has been terminated, you may be forced to reset your computer in order to reload COMMAND.COM.

Message: **Invalid country code**

Explanation: The country number in your CONFIG.SYS file was not found in the table of files.

Message: **Invalid date**

Explanation: The date you entered was not in the proper format. The date must be entered in the following format

m-dd-yy

where
mm is one or two numbers to indicate the month
dd is one or two numbers to indicate the day of the month
yy is one or two numbers to indicate the year

Message: **Invalid device**

Explanation: The device specified was not either CON, NUL, AUX, or PRN.

Message: **Invalid directory**

Explanation: DOS is unable to find the directory you specified. Either it does not exist or else the pointers to the name and location of the directory are not accessible. Make sure you entered the directory name correctly.

Message: **Invalid disk change reading drive (d:)**

Explanation: An error occurred while DOS was reading from a device.

Message: **Invalid disk change writing drive (d:)**

Explanation: An error occurred while DOS was writing to a device.

Message: **Invalid drive in search path**

Explanation: The drive you specified does not exist.

Message: **Invalid drive name or file**

Explanation: DOS can't find the drive or file name you specified.

Message: **Invalid drive specification**

Explanation: DOS can't find the drive you specified.

Message: **Invalid number of parameters**

Explanation: You have specified the wrong number of options as part of a DOS command entry.

Message: **Invalid parameter**

Explanation: You have specified an incorrect option as part of a DOS command entry.

Message: **Invalid path or file name**

Explanation: You have specified an incorrect path name or filename as part of a DOS command entry.

Message: **Invalid path, not directory, or directory not empty**

Explanation: While attempting to remove a directory, either you have specified an incorrect pathname or the pathname you entered indicates a directory that is not empty. Before a directory can be removed, you must erase all files from it and delete all subdirectories within it.

Message: **Invalid sub-directory entry**

Explanation: DOS cannot access the name of the subdirectory you specified. This usually indicates that the subdirectory does not exist. Make sure you entered the correct subdirectory name.

Message: **Invalid time**

Explanation: The time you entered was not in the proper format. The time must be entered in the following format

hh:mm:ss

where
hh is one or two numbers to indicate the hour
mm is one or two numbers to indicate the minutes past the hour
ss is one or two numbers to indicate the seconds past the minute

Message: **Label not found**

Explanation: When processing a batch file, DOS has encountered a batch file line that includes a GOTO command that refers to a label that cannot be found in the file. Check the batch file to make sure the label in the GOTO command EXACTLY matches the label. Make sure the label begins with a colon (:).

Message: **Memory allocation error**
Cannot load DOS, system halted

Explanation: You will have to reset or restart your computer. This error may indicate a problem with your copy of DOS; if so, you may have to recopy the DOS system files to the working disk.

Message: **No room for system on destination disk**

Explanation: There is not sufficient space on the disk you specified to copy the DOS system files. In order to copy system files to disk, the disk must be prepared using the /S option of the FORMAT command.

Message: **No room in directory for file**

Explanation: When you attempted to create a new file, the internal DOS directory of files was found to be full. You may be attempting to create too many files on the disk. A single-sided diskette is limited to 64 files. A double-sided diskette is limited to 112 files. A double-density disk is limited to 224 files. A fixed disk is limited to 512 files.

Message: **Not enough memory**

Explanation: There is not enough system memory (RAM) for DOS to run the command you specified.

Message: **Not ready error reading drive (d:)**

Explanation: An error occurred while DOS was reading from a drive. Make sure the diskette is in the drive and the drive door is closed.

Message: **Not ready error writing drive (d:)**

Explanation: An error occurred while DOS was writing to a drive. Make sure the diskette is in the drive and the drive door is closed.

Message: **Out of environment space**

Explanation: While using the SET command you have attempted to insert data into the environment. There is not enough room in the program environment to accept more data.

Message: **Path not found**

Explanation: As part of a DOS command you have specified an invalid path-name. Check the command syntax to make sure the path designation refers to an existing drive and directory.

Message: **Probable non-DOS disk**
Continue (Y/N)?

Explanation: The disk you specified is not recognized by DOS. The disk either was created by another system or with a version of FORMAT that is not supported on this version of DOS. Unfortunately, this message may also be displayed when you are using a disk that you know was prepared using the proper version of DOS. This can indicate that the internal indicators used by DOS to recognize the disk format have been destroyed; in this case, the disk is not usable and must be re-formatted.

If you are using a fixed disk when this message is displayed, you may be able to resolve the problems by running the CHKDSK command with the /F option.

Message: **Processing cannot continue**

Explanation: There is not enough system memory (RAM) to run the CHKDSK program.

Message: **Program too big to fit memory**

Explanation: The program you are attempting to run utilizes memory and there is not enough system memory (RAM) to load it. You must free-up some system memory before you can run the program. In order to do this, you may have to reset or restart your computer.

It is also possible that the program you are trying to run requires more memory than is available in your computer.

Message: **Read fault error reading drive (d:)**

Explanation: An error occurred while DOS was reading from a drive.

Message: **Sector not found error reading drive (d:)**

Explanation: An error occurred while DOS was reading from a drive. This error usually indicates that a part of the disk contains data that cannot be read. Other files on the disk may or may not also be damaged.

Message: **Sector not found error writing drive (d:)**

Explanation: An error occurred while DOS was writing to a drive. This error usually indicates that a part of the disk has become damaged or contains data that cannot be accessed by DOS. Other files on the disk may or may not also be damaged.

Message: **Sector size too large in file (filename)**

Explanation: The specified device driver used by CONFIG.SYS uses a sector size that is too large.

Message: **Seek error reading drive (d:)**

Explanation: An error occurred while DOS was reading from a drive. This error usually indicates that a part of the disk contains data that cannot be read. Other files on the disk may or may not also be damaged.

Message: **Seek error writing drive (d:)**

Explanation: An error occurred while DOS was writing to a drive. This error usually indicates that a part of the disk contains data that cannot be written to, either because of damage to the disk or because DOS cannot find markers on the disk that indicate the location to be written to. Other files on the disk may or may not also be damaged.

Message: **Track 0 bad - disk unusable**

Explanation: The FORMAT command encountered defective sectors on the beginning sectors of the disk (in the area used for system information). DOS can use a disk with bad sectors in any area except in the track 0 area.

Because of undocumented errors in the way DOS begins the FORMAT process, this message may not indicate, in all cases, that the disk is bad. Device signals from the drive may not be interpreted correctly by DOS. If this message is occasionally displayed when you begin to format the disk, try canceling the FORMAT command and trying it again. It may even be necessary to re-start the computer to clear the internally set error flag.

Message: **Unable to create a directory**

Explanation: DOS could not create the directory you specified. Check to see if there is already a file or directory by that name on the disk. Check to make sure there is room on the specified disk for another directory (a small amount of disk space is required for each directory your create).

Message: **Unrecognized command in CONFIG.SYS**

Explanation: DOS cannot recognize an entry in the CONFIG.SYS file. Make sure you have entered each line of the CONFIG.SYS file accurately and make sure there are no extraneous characters in the file.

Message: **Unrecoverable error in directory**
Convert directory to file (Y/N)?

Explanation: CHKDSK has encountered an error in checking a directory. If you respond to this message with **Y**, CHKDSK will convert the directory into a file so you can determine if it can be fixed.

Appendix B
Country-Specific Information

A number of capabilities related to country-specific DOS tasks were added to the DOS command set in version 3 and later. This appendix summarizes the country-specific features that are provided with the DOS versions.

Using Code Pages

DOS provides language support for the command set by using language-specific code pages. Several of the DOS commands are related to code page switching.

A code page is actually a table of characters (a character set) related to a specific country. DOS uses a code page to translate characters before they are displayed on the screen. Each character set consists of up to 256 characters. The character set contains all the special characters and accents used to display messages in a country's language.

DOS 3.3 provides support for five code pages (character sets). They are:

437 United States This is the default code page. It support all the standard english letters and numbers and special characters commonly found in standard ASCII tables.

850 Multilingual This is a "standard" set of 256 characters designed to support most of the world's Romantic languages (European, North American, and South American).

860 Portuguese Portuguese language support.

863 French-Canadian French-Canadian version of French language support.

865 Nordic Norwegian and Danish language support.

NOTE:
Printer support for the full character sets is provided by DOS; however, a graphics printer is required to display many of these characters.

Language Support Codes

DOS uses a "country code" to prepare and assign code pages. 19 country codes are available as listed below. The default country code is always 001 (United States) unless reset.

Country	Country Code	Code Pages	Keyboard
United States	001	437, 850	US
French-Canadian	002	863, 850	CF
Latin American	003	437, 850	LA
Netherlands	031	437, 850	NL
Belgium	032	437, 850	BE
France	033	437, 850	FR
Spain	034	437, 850	SP
Italy	039	437, 850	IT
Switzerland	041	437, 850	SF, SG
United Kingdom	044	437, 850	UK
Denmark	045	850, 865	DK
Sweden	046	437, 850	SV
Norway	047	850, 865	NO
Germany	049	437, 850	GR
English	061	437, 850 -	—
Portugal	351	850, 860	PO
Finland	358	437, 850	SU
Arabic	785	437, 850 -	—
Israel	972	437, 850 -	— -

Installing Code Page Switching

DOS 3.3 provides a way to switch from your existing code page to a new multi-national code page (to use the new character set).

To install the new code page you will need to create an "AUTOEXEC.BAT" file and a "CONFIG.SYS" file in the root directory of your start-up disk (if they do not already exist). If they already exist, you can modify them to configure the country-specific requirements.

The AUTOEXEC.BAT file should contain the "CP PREPARE" form of the MODE command appropriate for the code pages you want to use. Refer to the "MODE" command sections in Chapters 5 and 6 for more information.

The CONFIG.SYS file should contain references to the devices and .SYS files related to the international character sets. Refer to the "CONFIG.SYS" section in Chapter 5 for more information. You will need to include country-specific information for the DEVICE command and the COUNTRY command.

NOTE:

The COUNTRY.SYS file contains information regarding country-specific issues such as the format for the date and time, as well as other language-related methods of presentation and format. The KEYBOARD.SYS file contains country-specific tables used to direct the KEYB command while scan codes are converted to ASCII characters for output (refer to the KEYB command). However, none of these capabilities are related to the translation of text into a foreign language: DOS provides no capabilities for translating text to or from a foreign language.

Country-Specific Commands

The following DOS commands have related country-specific settings:

CHCP
KEYB
MODE
NLSFUNC
SELECT

For more information on using the country-specific aspects of these commands refer to each command's reference section in Chapter 6, "DOS Command Reference."

Appendix C
ASCII Character Code Chart

ASCII Value	Character	Control Character	ASCII Value	Character
000	(null)	NUL	032	(space)
001	☺	SOH	033	!
002	☻	STX	034	"
003	♥	ETX	035	#
004	♦	EOT	036	$
005	♣	ENQ	037	%
006	♠	ACK	038	&
007	(beep)	BEL	039	'
008	◘	BS	040	(
009	(tab)	HT	041	)
010	(line feed)	LF	042	*
011	(home)	VT	043	+
012	(form feed)	FF	044	,
013	(carriage return)	CR	045	-
014	♫	SO	046	.
015	☼	SI	047	/
016	►	DLE	048	0
017	◄	DC1	049	1
018	↕	DC2	050	2
019	!!	DC3	051	3
020	¶	DC4	052	4
021	§	NAK	053	5
022	▬	SYN	054	6
023	↨	ETB	055	7
024	↑	CAN	056	8
025	↓	EM	057	9
026	→	SUB	058	:
027	←	ESC	059	;
028	(cursor right)	FS	060	<
029	(cursor left)	GS	061	=
030	(cursor up)	RS	062	>
031	(cursor down)	US	063	?

Appendix C. ASCII Character Code Chart

ASCII Value	Character	ASCII Value	Character
064	@	096	`
065	A	097	a
066	B	098	b
067	C	099	c
068	D	100	d
069	E	101	e
070	F	102	f
071	G	103	g
072	H	104	h
073	I	105	i
074	J	106	j
075	K	107	k
076	L	108	l
077	M	109	m
078	N	110	n
079	O	111	o
080	P	112	p
081	Q	113	q
082	R	114	r
083	S	115	s
084	T	116	t
085	U	117	u
086	V	118	v
087	W	119	w
088	X	120	x
089	Y	121	y
090	Z	122	z
091	[	123	{
092	\	124	¦
093	]	125	}
094	^	126	~
095	_	127	⌂

ASCII Value	Character	ASCII Value	Character
128	Ç	160	á
129	ü	161	í
130	é	162	ó
131	â	163	ú
132	ä	164	ñ
133	à	165	Ñ
134	å	166	ª
135	ç	167	º
136	ê	168	¿
137	ë	169	⌐
138	è	170	¬
139	ï	171	½
140	î	172	¼
141	ì	173	¡
142	Ä	174	«
143	Å	175	»
144	É	176	▒
145	æ	177	▒
146	Æ	178	▒
147	ô	179	│
148	ö	180	┤
149	ò	181	╡
150	û	182	╢
151	ù	183	╖
152	ÿ	184	╕
153	Ö	185	╣
154	Ü	186	║
155	¢	187	╗
156	£	188	╝
157	¥	189	╜
158	Pt	190	╛
159	ƒ	191	┐

ASCII Value	Character	ASCII Value	Character
192	└	224	α
193	┴	225	ß
194	┬	226	Γ
195	├	227	π
196	─	228	Σ
197	┼	229	σ
198	╞	230	µ
199	╟	231	τ
200	╚	232	Φ
201	╔	233	Θ
202	╩	234	Ω
203	╦	235	δ
204	╠	236	∞
205	═	237	Ø
206	╬	238	ε
207	╧	239	∩
208	╨	240	≡
209	╤	241	±
210	╥	242	≥
211	╙	243	≤
212	╘	244	⌠
213	╒	245	⌡
214	╓	246	÷
215	╫	247	≈
216	╪	248	°
217	┘	249	∙
218	┌	250	·
219		251	√
220		252	n
221		253	²
222		254	■
223		255	(blank 'FF')

Index

Advanced commands 5 - 37 to 5 - 60
MODE 5 - 38
PROMPT 5 - 46
SET 5 - 56
XCOPY 5 - 58
Advanced users 5 - 1 to 5 - 60
Alt key scan code 5 - 51
APPEND command 3 - 10, 6 - 12
options 3 - 10
ASCII character table C - 1
ASCII characters 5 - 2
producing from keyboard 5 - 2
ASSIGN command 6 - 15
Asynchronous communications
settings 5 - 43
ATTRIB command 6 - 17
AUTOEXEC batch file 4 - 3
example 4 - 34

Backing up files 6 - 20
BACKUP command 6 - 20
Batch file commands 4 - 9
CALL 4 - 9
ECHO 4 - 10
FOR 4 - 11
GOTO 4 - 13
IF 4 - 15
SHIFT 4 - 19
PAUSE 4 - 20
REM (Remark) 4 - 21
Batch file examples 4 - 22 to 4 - 37
backup files 4 - 26
change directories 4 - 23
create option menu 4 - 31
display a directory listing 4 - 23
powerful AUTOEXEC.BAT
4 - 33
protect fixed disk 4 - 25
Batch files 4 - 1
advanced techniques 4 - 22
creating 4 - 2
creating GET.COM 4 - 27
definition 4 - 1
interactive batch files 4 - 27
interrupting 4 - 2
passing parameters 4 - 5
BREAK command 5 - 20, 6 - 23
Buffer setting 5 - 20
BUFFERS command 5 - 20

CALL command 4 - 9, 6 - 24
Calling batch files 4 - 9
Changing code page 6 - 25
Changing directories 3 - 7, 6 - 27
Changing drives 1 - 4
Changing file names 2 - 20, 6 - 96
Changing input/output device 6 - 41
CHCP command 6 - 25
CHDIR (CD) command 3 - 7, 6 - 27
Check for Ctrl/Break 5 - 20, 6 - 23
Check for Ctrl/C 5 - 20
Checking disk status 2 - 2, 6 - 29
CHKDSK command 1 - 8, 2 - 2, 6 - 29
error messages 2 - 3 to 2 - 5
Clear screen display 6 - 33
CLS (Clear Screen) command 6 - 33
Code page selection 5 - 45
Color of display selection 5 - 53

Combining files 2 - 7
COM extension 1 - 9
COMMAND command 6 - 34
Command entry instructions 1 - 12
drive designator 1 - 12
notation rules 1 - 13
path designator 1 - 12
switches 1 - 12
use of space character 1 - 12
Command piping 5 - 15
examples 5 - 15 to 5 - 18
Command summary 6 - 5 to 6 - 11
COMP command 6 - 36
Comparing disks 6 - 46
Comparing files 6 - 36
Conditional operations 6 - 66
CONFIG.SYS file 5 - 19
Configuration commands 5 - 20 to 5 - 25
BREAK 5 - 20
BUFFERS 5 - 20
COUNTRY 5 - 21
DEVICE 5 - 22
FCBS 5 - 22
FILES 5 - 23
LASTDRIVE 5 - 24
SHELL 5 - 24
STACKS 5 - 25
Configuring the system 5 - 19
Control key scan code 5 - 51
Converting files to binary format 6 - 49
Converting lost chains 2 - 3
COPY command 2 - 5, 6 - 38
combining files 2 - 7
copying with new filename 2 - 6
options 2 - 7, 6 - 39
Copying directories 6 - 122
Copying disks 2 - 13, 6 - 47
Copying files 2 - 5, 6 - 122
combining files 2 - 7
using new name 2 - 6
using pathnames 2 - 6
XCOPY 5 - 58, 6 - 122
Copying subdirectories 6 - 122
COUNTRY command 5 - 21
Country-specific information B - 1
Country-specific settings 5 - 21
CTTY command 6 - 41

DATE command 2 - 8, 6 - 42
DEL (ERASE) command 2 - 11, 6 - 43
recovering deleted files 2 - 12
Deleting files 2 - 11, 6 - 43
DEVICE command 5 - 22
Device driver setting 5 - 22
Device names 5 - 7
DIR command 2 - 9, 6 - 44
options 2 - 9
Directories 3 - 1
creating directories 3 - 1
displaying directory tree 3 - 3, 3 - 11
DISKCOMP command 6 - 46
DISKCOPY command 2 - 9, 6 - 47
Diskette cautions 1 - 21
labeling 1 - 21
Disk partitions 3 - 13
Disk storage capacity 1 - 24
Disk types 1 - 22
Display contents of environment 5 - 56
Display contents of file 6 - 118
Display current date 6 - 42
Display current time 6 - 115
Display differences between files 6 - 53
Display directory of files 2 - 9, 6 - 44
Display directory paths 6 - 117
Display disk volume label 6 - 120
Display DOS version number 6 - 119
DOS command reference 6 - 1 to 6 - 123

DOS command summary 6 - 5 to 6 - 11
DOS editing keys 5 - 4
DOS error messages 1 - 25, A - 1
DOS functions 1 - 2
DOS program types 1 - 5
  internal 1 - 5
  external 1 - 8
DOS prompt 1 - 3
  changing 5 - 46
  character options 5 - 47
DOS version number 6 - 119
Drive designator 1 - 12, 1 - 13, 1 - 15

ECHO command 4 - 10, 6 - 48
Erasing files 6 - 43
Error messages 1 - 25, A - 1
.EXE extension 1 - 9
EXE2BIN command 6 - 49
EXIT command 6 - 51
Exit secondary command processor 6 - 51
External programs 1 - 8

FASTOPEN command 6 - 52
FC command 6 - 53
FCBS command 5 - 22
FDISK command 6 - 55
File control block setting 5 - 22
Filename extensions 1 - 14, 1 - 18
Filenames 1 - 14, 1 - 15, 1 - 16
  legal characters 1 - 16
  reserved characters 1 - 17
  useful filenames 1 - 19
  use of upper and lower-case 1 - 17
  valid names 1 - 17
  wildcards 1 - 19
FILES command 5 - 23
File locking 6 - 106
File sharing 6 - 106
Filter commands 5 - 8
  FIND 5 - 8
  MORE 5 - 8
  SORT 5 - 8
FIND command 6 - 56
Find specific string of text 6 - 56
Fixed disk commands 3 - 4
  APPEND 3 - 10
  CHDIR (CD) 3 - 7
  MKDIR (MD) 3 - 4
  PATH 3 - 8
  RMDIR (RD 3 - 6
  TREE 3 - 11
Fixed disk setup 3 - 12
FOR command 4 - 11, 6 - 58
FORMAT command 2 - 12, 6 - 59
  options 2 - 14
Format notation 1 - 10
Formatting disks 2 - 12, 6 - 59
  with country-specific information 6 - 102
Formatting fixed disks 3 - 12
Frequently used commands 2 - 2 to 2 - 23
  CHKDSK 2 - 2
  COPY 2 - 5
  DATE 2 - 8
  DIR 2 - 9
  DISKCOPY 2 - 9
  DEL (ERASE) 2 - 11
  FORMAT 2 - 12
  PRINT 2 - 16
  RENAME (REN) 2 - 20
  TIME 2 - 21
  TYPE 2 - 22
Function key scan codes 5 - 51

GOTO command 4 - 13, 6 - 61
GRAFTABL command 6 - 62

GRAPHICS command 5 - 37, 6 - 64

High level format 3 - 13

IF command 4 - 15, 6 - 66
Insert remarks in batch file 6 - 95
Internal programs 1 - 5
  directory management commands 1 - 5
  file management commands 1 - 5
  general purpose commands 1 - 6

JOIN command 6 - 70

KEYB command 6 - 72
Keyboard support selection 6 - 72

LABEL command 6 - 74
LASTDRIVE command 5 - 24
Lost clusters 2 - 3
Low level format 3 - 13

Making directories 3 - 4, 6 - 75
Maximum number of drives setting 5 - 24
Maximum number of files setting 5 - 23
Microsoft 1 - 2
MKDIR (MD) command 3 - 4, 6 - 75
MODE command 5 - 38, 6 - 76
  asynchronous communications settings 5 - 43, 6 - 79
  code pages 5 - 45, 6 - 81
  monitor settings 5 - 41, 6 - 78
  parallel printer output 6 - 81
  printer settings 5 - 39, 6 - 76
Monitor display settings 5 - 41, 6 - 78
MORE command 6 - 84

NLSFUNC command 6 - 86
Non-contiguous blocks 2 - 5
Notation conventions 1 - 11
  special characters 1 - 11
  use of ellipses 1 - 14
  use of square brackets 1 - 14
  use of vertical bar 1 - 14
Notation rules 1 - 13
Number of stack frames setting 5 - 25

Operating systems 1 - 1

Parameters 6 - 107
Partitioning the disk 3 - 13
Passing parameters 4 - 5
PATH command 3 - 8, 6 - 87
Path designator 1 - 12, 1 - 13, 1 - 15
PAUSE command 4 - 20
Piping 5 - 8
Prepare fixed disk 6 - 55
PRINT command 2 - 16, 6 - 89
  error messages 2 - 19
  options 2 - 17, 5 - 27
Print data files 6 - 89
Print from graphics screen 6 - 64
Printing with DOS 2 - 16, 5 - 26
  batch file printer program 5 - 31
  characters per line setting 5 - 39
  continuous retry setting 5 - 40
  laser printers 5 - 36
  lines per inch setting 5 - 40
  other printing methods 5 - 27
  PRINT program options 5 - 26

PROMPT command 5 - 34
sample printer program (BASIC) 5 - 28
special printer capabilities 5 - 27
Programming function keys 5 - 51
Prompt character options 5 - 47
PROMPT command 5 - 34, 5 - 46, 6 - 92
character options 5 - 47, 6 - 92
display colors selection 5 - 53
display date and time 5 - 49
example sequence 5 - 48
function key scan codes 5 - 51
programming keys 5 - 49
screen control 5 - 53

RECOVER command 6 - 94
Recovering deleted files 2 - 12
Redirecting input and output 5 - 9
examples 5 - 9
Redirection and piping 5 - 8
Reload DOS command processor 6 - 34
REM (Remark) command 4 - 21, 6 - 95
Removing directories 3 - 6, 6 - 101
RENAME (REN) command 2 - 20, 6 - 96
Renaming files 2 - 20
Repeat execution of commands 6 - 58
REPLACE command 6 - 97
Replace stored files 6 - 97
RESTORE command 6 - 99
Restoring backed up files 6 - 99
RMDIR (RM) command 3 - 6, 6 - 101
Root directory 3 - 2

Sample format notation 1 - 10
Scan codes 5 - 51
Sector problems 6 - 94
SELECT command 6 - 102
Send output to console 6 - 84
SET command 5 - 56, 6 - 104
Set system date 2 - 8, 6 - 42
Set system time 2 - 21
Set path 3 - 8
SHARE command 6 - 106
SHELL command 5 - 24
SHIFT command 4 - 19, 6 - 107
Shift key scan code 5 - 51
SORT command 6 - 109
Sorting input 6 - 109
Spaces in syntax 1 - 12
Special command syntax characters 1 - 11
STACKS command 5 - 25
Startup display 1 - 3
Subdirectories 3 - 2
SUBST command 6 - 111
Switches 1 - 12, 1 - 14, 1 - 16
Syntax notation 6 - 1
command usage 6 - 3
syntax elements 6 - 2
SYS (System) command 6 - 113
System configuration 5 - 19

TIME command 2 - 21, 6 - 115
TREE command 3 - 11, 6 - 117
Turn batch file display on/off 6 - 49
TYPE command 2 - 22, 6 - 118
ctrl/prtsc option 2 - 23
shift/prtsc option 2 - 23

Unconditional branch 6 - 61

VER command 6 - 119
VERIFY command 6 - 120
Verifying data written to disk 6 - 120
Virtual drives 6 - 111
VOL (Volume) command 6 - 121
Volume labels 6 - 74

Wildcards 1 - 19
    asterisk 1 - 20
    questionmark 1 - 19

XCOPY command 5 - 58, 6 - 122
    options 5 - 58, 6 - 124
    special rules 5 - 59

NOTES

NOTES

DOS the *EASY* Way

A Complete Guide to Microsoft's MS-DOS

Includes all commands and functions for
Version 3.3 of MS-DOS and PC-DOS.

This comprehensive guide to today's most popular operating system for personal computers is designed for all levels of users. It contains straight-forward explanations of the basic operating commands as well as a step-by-step introduction to the tricks and techniques used by the experts.

The book provides a large number of carefully-explained examples that show you exactly how to carry out your everyday computer tasks more efficiently.

A complete reference guide provides the command syntax, options, and usage of every MS-DOS command currently in use (includes commands for version 3.3 of both MS-DOS and PC-DOS).

Explanations of how each command and function can be put to practical use is provided for both beginning and advanced users. With this book, even the most complex commands can become valuable everyday tools. You will quickly learn how to:

- **Use internal and external DOS commands**
- **Manage disks, files and directories**
- **Organize your fixed-disk**
- **Take advantage of hidden DOS capabilities**
- **Create batch files to automate repetitive commands**
- **Use the DOS configuration and system files**
- **Design menu programs using DOS commands**
- **Enhance the DOS printing programs**

ABOUT THE AUTHOR Dr. Murdock is an Associate Professor at California State University, Long Beach where he teaches courses on educational computing and instructional design. He is the author of a number of books about using computers, including the Home and School Guide to the Apple Macintosh and the Home and School Guide to IBM-Compatible Personal Computers.

ISBN 0-923178-00-7

H.O.T. Press