Chapter 2

Operating the CBM Computer

This chapter explains how you use the keyboard to operate the computer, disk drive unit, cassette unit, and printer.

You tell a CBM computer what to do using "statements." Statements are instructions to the computer. The computer "executes" statements in order to do their bidding. The CBM computer understands statements written in CBM BASIC (Beginning All-purpose Symbolic Instruction Code).

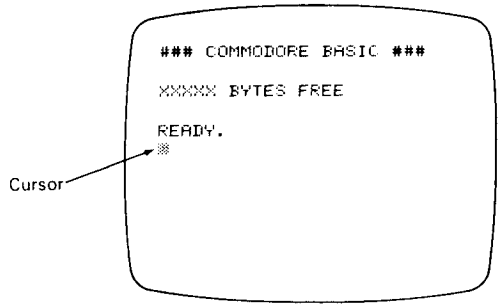
When you enter a BASIC statement, the statement line can be up to 80 characters in length. Eighty characters is equal to two lines on the screen display for the CBM graphic and business model computers, and one display line for the CBM 8000 series computer. When entering a statement line, type in the characters and terminate the line by pressing the RETURN key. If you type in 40 or more characters on the small-screen CBM computer without pressing the RETURN key, the cursor automatically drops down to the next display line, and you can continue entering up to 80 characters. If you continue typing past the 80th character on either the small- or the large-screen CBM computer, the CBM computer will respond with a ?SYNTAX ERROR message when you press the RETURN key.

You must press the RETURN key in order to terminate every BASIC statement. The CBM computer uses the RETURN character as a signal that the line is complete and ready to be analyzed.

BASIC statements may be entered in two ways: immediate mode or program mode. Immediate mode statements are executed immediately: hence the name "immediate mode." These statements are generally short, and are *not* stored in the computer's memory. Statements entered in program mode *are* stored in computer memory. But program mode statements are not executed until you explicitly instruct the computer to do so.

IMMEDIATE MODE

When powered up, the CBM computer is in immediate mode. Immediate mode is indicated by a flashing cursor on the screen.



The computer remains in immediate mode until you execute a BASIC program. After the program has executed, the computer returns to immediate mode.

KEYBOARD INPUT IN IMMEDIATE MODE

The flashing cursor is displayed on the screen at the character position where the next character input from the keyboard will appear. As each character entered appears on the screen, the cursor advances one space to the right and waits for the next character, as shown in Figure 2-1. Some keys move the cursor without displaying any character. If a CURSOR key is pressed, the cursor moves one space in the direction of the arrow on the CURSOR key and waits for further input. Pressing the RETURN key drops the cursor down to the first position on the line below.

You can input any sequence of characters via the CBM computer keyboard, but in immediate mode the CBM computer assumes any input to be part of a BASIC statement and interprets the input by the rules of the CBM BASIC language. That is, the CBM computer tries to interpret all input as BASIC statements.

Figure 2-2 illustrates valid statement input in immediate mode. After the RETURN key is pressed, the statement is executed immediately, and results (if any) are displayed on the line below. Non-statement entry is shown in Figure 2-3. In the first two screens, upon pressing the RETURN key the display responds with a ?SYNTAX ERROR message.

A ?SYNTAX ERROR message signals that the CBM computer cannot interpret the input as a valid BASIC statement. If you are not trying to enter a statement, simply ignore the syntax error message. An exception is shown in the third screen of Figure 2-3. Graphic characters and cursor control characters do not generate a syntax error *unless mixed with* alphabetic or numeric characters. This allows you to "draw" directly onto the screen using graphic symbols.

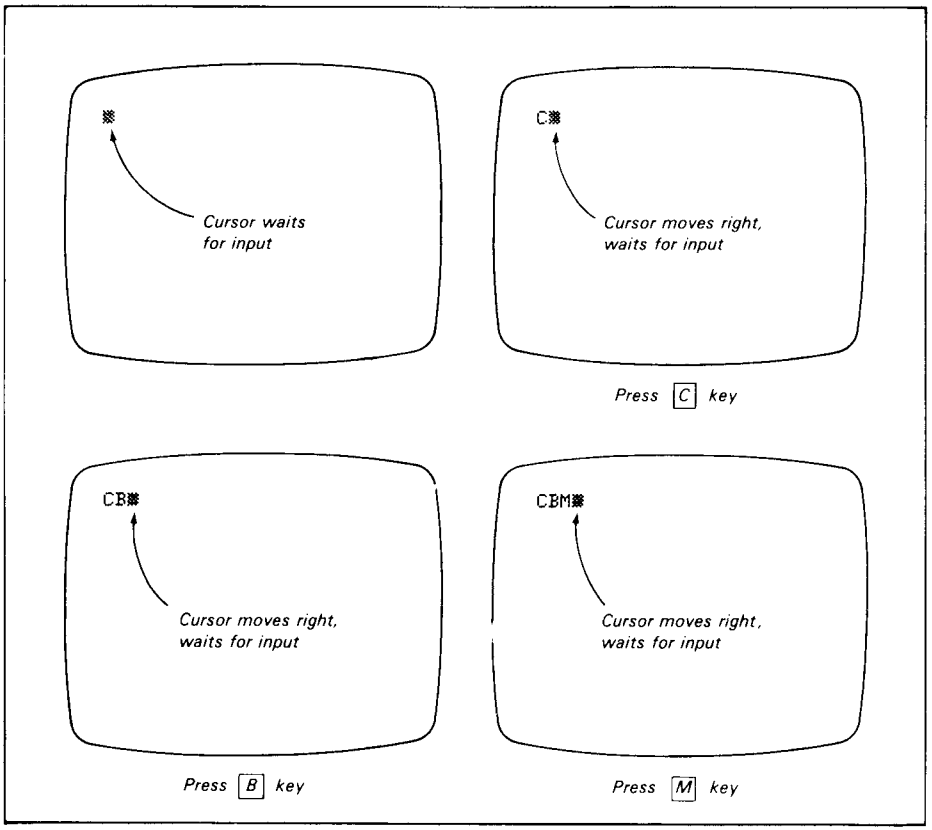


Figure 2-1. Keyboard Input in Immediate Mode

Statements in Immediate Mode

Statements entered in immediate mode do not, and cannot, **begin with line numbers**. Later we will examine exactly what you can include in an immediate mode statement. For the discussion at hand you only need to know how to input a few elementary statements, therefore we will explain how to write statements that print characters, solve arithmetic equations, and move the cursor.

The PRINT Statement

$$\left\{\begin{matrix}\text{PRINT} \\ ?\end{matrix}\right\} \text{data}$$

The PRINT statement shown above is the most frequently used in immediate mode statements. The PRINT statement instructs the computer to display data on the screen. Upon pressing the RETURN key, the display appears on the line below the PRINT statement. If the word PRINT (or its abbreviation '?') is not placed before the data item to be displayed, no display will appear on the screen.

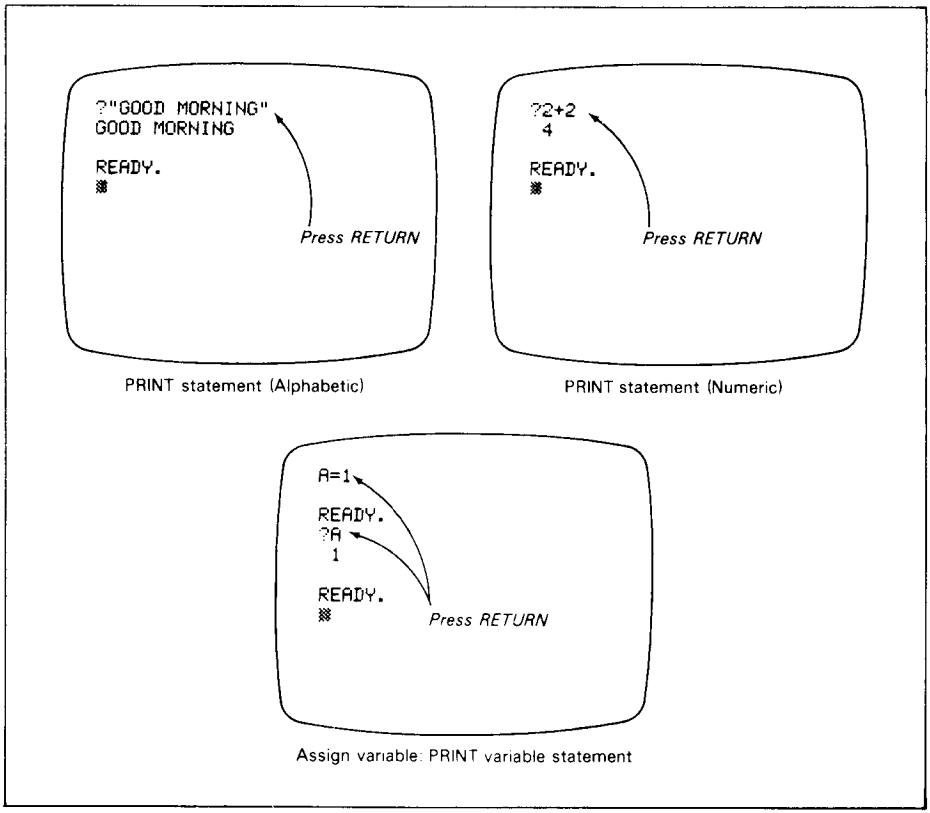


Figure 2-2. Valid Immediate Mode Statement Input

Printing Strings

CBM computers can recognize simple sequences of characters that have no "special" meaning. The word "string" is the computer jargon that describes such text.

A string is a sequence of one or more characters enclosed in double quotation marks. Here are some examples of strings:

```
"HI!"
"SYNERGY"
"12345"
"10.44 IS THE AMOUNT"
"22 UNION SQUARE, SAN FRANCISCO, CA"
```

All data keys (alphabetic, numeric, special symbols, and graphics), as well as the cursor control keys and the REVERSE ON/OFF key, can be included in a string. The only keys that cannot be used within a string are RUN/STOP and RETURN.

A string may be displayed, directly or indirectly, by assigning the string to a "string variable" and then printing the contents of the string variable.

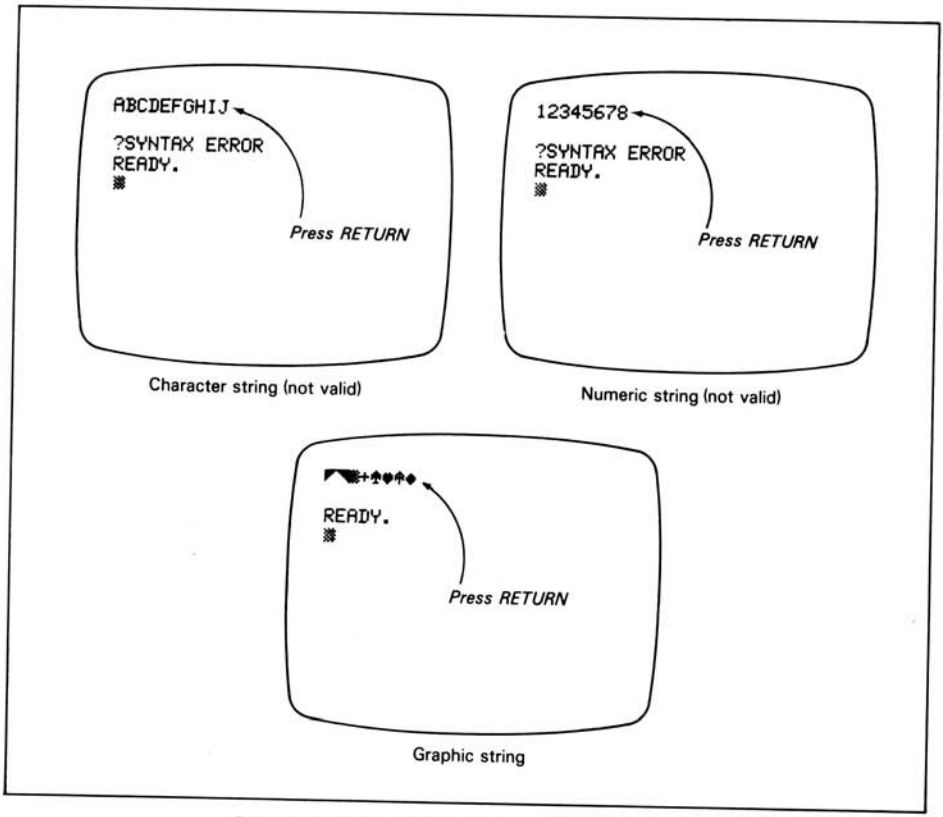


Figure 2-3. Non-statement Immediate Mode Input

To print a string directly, key in PRINT (or its abbreviation '?') followed by the string to be printed. The string must be enclosed within quotation marks.

PRINT "string"

where:

string is a string of text

Upon pressing the RETURN key, the string is displayed on the following line. Some examples are shown below. The shaded characters designate your input, unshaded characters are the computer's response.

```
PRINT "MONDAY"
MONDAY

READY.

PRINT "MAY 12, 1980"
MAY 12,1980

READY.

PRINT "12345"
12345

READY.
```

Try inputting a couple of your own PRINT statements to display strings. When you press the RETURN key, if you get a ?SYNTAX ERROR message, you probably misspelled PRINT. If you get a zero (0) answer instead of your string, you have forgotten the set of quotation marks that precede the string. Keep trying until you can successfully print a string in immediate mode.

A string is printed indirectly by assigning the string to a "string variable," and then printing the contents of the variable. A string variable is an identifier, or a name, that can represent any text string. A string variable's identifier, or name, has a letter, optionally followed by another letter or a number. A "$" character must end the string variable name. Here are some examples of string variable names:

```
A$
M1$
F6$
```

You need two statements to print a string indirectly. The first statement assigns the string to a variable name as follows:

```
V$ = "string"
```

where:

V$	is the string variable name
string	is a string text

A PRINT statement comes next; it displays the contents of the string variable on the screen:

```
PRINT V$
```

where:

V$	is the string variable name assigned to a string of text

Here are a few examples:

```
A$="TAKE THE PROGRAM AND RUN!"

READY.
PRINT A$
TAKE THE PROGRAM AND RUN!

READY.
DY$="TUESDAY"

READY.
PRINT DY$
TUESDAY

READY.
B1$="♠ ♥ ♦ ♣"

READY.
PRINT B1$
♠ ♥ ♦ ♣

READY.
```

Try telling the CBM computer to print your name indirectly, using a variable name. Type in a string name (don't forget the dollar sign!), an equals sign, and your name enclosed in quotation marks. Press the RETURN key, as shown below:

```
NM$="JIMMY OLSON"  <-- RETURN key pressed

READY.
```

If you get a ?TYPE MISMATCH ERROR message, either your name is not enclosed in quotation marks, or the string variable name is incorrect (missing a dollar sign?). Now, type PRINT, followed by the string variable name. Press the RETURN key. Your name should be displayed on the line below:

```
NM$="JIMMY OLSON"  <-- RETURN key pressed

READY.
PRINT NM$  <-------- RETURN key pressed
JIMMY OLSON

READY.
```

If something other than your name is displayed, there is a mistake in your PRINT statement. If a ?SYNTAX ERROR message appears, you probably misspelled the word PRINT. If you used the wrong string variable name, you will get no display, since the new name describes a variable that has no contents. If the dollar sign was omitted from the string variable name, a '0' will be displayed. If any of these errors occur, go back and try it again.

ARITHMETIC CALCULATIONS

CBM BASIC can handle arithmetic. Leaving the complicated stuff for Chapter 4, we will now look at some simple addition, subtraction, multiplication and division. An arithmetic equation appearing in a PRINT statement instructs the computer to solve an equation and display the result immediately.

If you put an arithmetic equation into the PRINT statement, 'PRINT' or '?' must precede the arithmetic equation which is to be evaluated. An equals sign *cannot* appear in the statement:

```
{PRINT}
{  ?  } equation
```

Here are some examples:

```
PRINT 2 + 2
PRINT 5/10
? 2.5
? (100/20) - 16.334
```

Upon pressing the RETURN key, the CBM computer calculates the equation and displays the answer.

Try an example: Type in PRINT or ?. Next, enter the arithmetic expression 2 + 2. Press the RETURN key. The answer should be displayed on the line below, as shown:

```
PRINT 2+2  <-------- Press RETURN key
 4

READY.
```

Below are some more examples you may want to try. Type in the shaded characters; the unshaded characters are the computer's response.

```
PRINT 2+2
 4

READY.
PRINT 5/10
 .5

READY.
? 2.5
 2.5

READY.
? (100/20)+16.334
 21.334

READY.
```

Arithmetic Calculations using Variables

Just as a string variable can represent a string, so a numeric variable can represent a number. A numeric variable's name has a letter, optionally followed by another letter or a number. Numeric variable names are *not* terminated by a dollar sign.

You must use two statements to print a number via a numeric variable. The first statement assigns the number to the numeric variable as follows:

```
V = n
```

where:

V	is the numeric variable name
n	is a number of numeric equation assigned to the variable

The second statement is a PRINT statement. It displays the value of the numeric variable on the screen:

```
PRINT V
```

where:

V	is the numeric variable name assigned to a number or numeric equation

Here are some sample statements assigning numbers to numeric variables:

```
A = 1
NM = 2.56
B1 = 1000/10
```

Notice that the numbers and equations are *not* enclosed within quotation marks.

Here are some examples of numeric assignment and PRINT statements:

```
C=100

PRINT C
 100

READY.
F1=1234.78

PRINT F1
 1234.78

READY.
```

CURSOR MOVEMENT

Cursor movement in immediate mode allows you to move the cursor and alter the screen display instantly. **When a cursor key is pressed, the cursor moves in the direction shown by the arrow on the key.** The cursor keys move the cursor around the screen, and over existing characters, without altering the display. These cursor keys include CURSOR HOME, CURSOR UP/DOWN, and CURSOR LEFT/RIGHT.

Cursor keys that move and alter the screen display include CLEAR SCREEN and INSERT/DELETE. CLEAR SCREEN wipes out the screen display and puts the cursor in the first character position of the top row; this is called the "home" position. INSERT/DELETE inserts spaces to the right of the cursor, or deletes the characters to the left of the cursor.

Whenever a cursor key is pressed after a quotation mark (or any *odd* number of quotation marks), the cursor movement is treated as part of a string. The cursor key is treated as a string character, and a special character appears in the string to represent that cursor movement, as shown in Table 2-1.

Table 2-1. String Representations of Cursor Keys

Function		Key	String Symbol
DELETE		INST DEL	(Reverse shifted T)
INSERT	Shifted	INST DEL	Not programmed
Home Cursor		CLR SCREEN HOME	(Reverse S)
Clear Screen	Shifted	CLR SCREEN HOME	(Reverse Shifted S)
Cursor Down		CURSOR	(Reverse Q)
Cursor Up	Shifted	CURSOR	(Reverse Shifted Q)
Cursor Right		CURSOR	(Reverse])
Cursor Left	Shifted	CURSOR	(Reverse Shifted])

Here are some examples of cursor movement keys appearing within character strings:

```
PRINT "1<CRSR→>2<CRSR→>3<CRSR→>4<CRSR→>5"
1 2 3 4 5

READY.
■

PRINT "♠<CRSR←><CRSR→>♥<CRSR←><CRSR→>♦<CRSR←><CRSR→>♠"
♠
♥
♦
♠

READY.
■

PRINT "*<CLR SCREEN><CRSR↓><CRSR←><CRSR↓><CRSR↓>*<CRSR↑>*<CRSR↑>*<CRSR↑>*
          <CRSR↑>*<CRSR↓><CRSR↓><CRSR↓><CRSR↓><CRSR↓>"
    *
   *
  *
 *
*

READY.
■
```

PROGRAM MODE

When you enter statements in program mode, they are stored in computer memory. Such statements must begin with a line number.

PROGRAM ENTRY

A program consists of one or more BASIC statements which, when executed in the proper sequence, cause the computer to perform a required task.

Programs may be entered via the keyboard. Programs may also be loaded into computer memory from an external peripheral (such as a cassette or diskette drive).

If entered from the keyboard, each statement is typed in with an initial line number. When the RETURN key is pressed at the end of the statement, the statement is stored in memory for later use, although it remains displayed on the screen. Program statements remain in memory until deleted, or until power is switched off. To avoid losing your program, you must save it either on cassette tape or on a diskette. Once the program has been saved on either medium, it can be loaded back into computer memory at any time.

At this point you need not know how to "write" a BASIC program; writing BASIC programs is the subject of Chapter 4. However, you now know enough about your CBM computer to **type in the short prewritten program called BLANKET, shown below.**

```
10 FOR I=1 TO 800
20 PRINT "A";
30 NEXT I
40 PRINT "PHEW!"
50 END
```

If you don't understand BASIC or this BLANKET program don't worry about it; you do not need to understand how or why it works. We will be using this program to illustrate computer operations.

When you type in this program, if you make a mistake, hit the RETURN key and then reenter the statement. If you get a ?SYNTAX ERROR message just retype the statement. Stop after you have typed the entire program correctly.

We will now look at how you can edit a program that is in the computer's memory.

Program LIST

The LIST statement displays the program which is currently in computer memory. After listing a program you can examine it for errors. The format for the LIST statement is:

LIST	(blank)	List entire program.
	line	List one line.
	$line_1$-$line_2$	List from $line_1$ to $line_2$.
	-line	List from start to line.
	line-	List from line to end.

where:

line	is a line number. $Line_1$ is a lower number than $line_2$. All line numbers are inclusive. The line numbers do not need to be actual line numbers in the program. In the case of listing one non-existent line, a blank line is printed.

If the program is longer than 23 lines, LIST will "scroll" the beginning lines of the program up off the screen. Use the line parameters in this case to display only the desired program lines.

Here are some examples:

LIST	List entire program.
LIST 50	List line 50.
LIST 60-100	List all lines in the program between lines 60 and 100, beginning with line 60 and ending with line 100.
LIST -140	List all lines in the program from the beginning of the program through line 140.
LIST 20000-	List all lines in the program from line 20000 to the end of the program.

List the program you entered previously. Type LIST, press RETURN, and your program should be displayed on the screen:

```
LIST

 10 FOR I=1 TO 800
 20 PRINT "A";
 30 NEXT I
 40 PRINT "PHEW!"
 50 END
READY.
```

Notice that the cursor disappears while the program is being listed on the screen, to reappear at the end of the program list at the bottom of the program, waiting in immediate mode for further input.

Now practice listing only a portion of the program. To display line number 10, type in LIST and the number 10. Press the RETURN key:

```
LIST 10

 10 FOR I=1 TO 800

READY.
```

To see line numbers 20 through 40 type in LIST 20-40:

```
LIST 20-40

 20 PRINT "A";
 30 NEXT I
 40 PRINT "PHEW!"
READY.
```

Practice listing the program in many different ways until you understand the LIST statement completely.

Program RUN

The RUN statement executes the program currently stored in computer memory. As soon as you enter the RUN statement and press the RETURN key the program executes. The format for the RUN statement is:

RUN	(blank)	Begin execution at the lowest-numbered line.
	line	Begin execution at the specified line.

Try executing the example program. Enter RUN, then press the RETURN key Sit back and watch. The screen should rapidly fill up with 800 A's beginning on the line directly below the RUN statement, and end with a PHEW!. Notice that while the program is executed, the CBM computer is in program mode and the cursor disappears.

When the program has completed its execution, the computer drops out of program mode and returns to immediate mode; a READY message is displayed and the cursor flashes at the bottom of the program output.

```
RUN
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
PHEW!

READY.
```

You can tell the computer to begin program execution at any line in the program by specifying the line number. For instance, you can tell the computer to execute the example program starting at line 40, as follows:

```
RUN 40
PHEW!

READY.
```

You should use caution when beginning program execution at a specified line. Often a program will not run correctly if it is not executed from the beginning. Were you to start the example program from line 20 or 30, the computer would respond with a syntax error because the program cannot begin execution on those lines.

Stopping Program Execution

A program may be stopped during execution. This is called a program "break." **A break will occur if you press the STOP key while the program is executing.** Following a break, a break message is displayed identifying the line at which the break occurred:

```
RUN
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAA <-------------------------- STOP key pressed
BREAK IN 40 <---------------------------- Break message displayed
READY.
```

During a program break the CBM computer returns to immediate mode allowing you to LIST, LOAD, SAVE, and VERIFY programs, and perform any immediate mode operations using CBM BASIC statements — including changing the values of program variables — before continuing program execution.

Continuing a Stopped Program

The "continue" statement, CONT, resumes program execution following a break. Execution continues at the exact point where the break occurred:

```
RUN
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAA <------------------------------ STOP key pressed
BREAK IN 40
READY.
CONT <--------------------------------------- CONT command entered
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAPHEW!

READY.
▒
```

CONT cannot continue program execution if an error occurred and an error message is displayed.

Deleting a Program from Memory

You can delete a program.

To delete program from computer memory, type in NEW and press the RETURN key. The current program is erased from computer memory, though it does not alter the screen display. Be careful when using the NEW statement; you may lose a program accidentally. Test the NEW statement now to delete the example program:

```
LIST <-------------------- LIST the program

 10 FOR I=1 TO 800
 20 PRINT "A";
 30 NEXT I
 40 PRINT "PHEW!"
 50 END
READY.
NEW <--------------------- Delete program from memory

READY.
LIST <-------------------- Re-LIST the program
     <-------------------- No program in memory
READY.
▒
```

Always erase an old program using the NEW statement before entering a new program. Otherwise you will end up with a hodgepodge of the new and old program statements.

STANDARD AND ALTERNATE CHARACTER SETS

Recall that every CBM computer has a "standard" and an "alternate" character set. These character sets are summarized in Appendix A for different models of CBM computers.

When powered up, PET models select the standard character set; CBM models select the alternate character set. To change character sets you change the value in memory location 59468. If 59468 is 12, the standard character set is selected. If 59468 is 14, the alternate set is selected. The following commands change location 59468:

```
POKE 59468,12   Activate the standard
                  character set
POKE 59468,14   Activate the alternate
                  character set
```

Programming with the Alternate Character Set

Never use the SHIFT key to select upper-case letters within BASIC statements when using the CBM computer alternate character set.

All CBM computers assume that the standard character set is present. When you press a key in shifted mode the interpreter assumes that the shifted standard character has been used.

This can cause you a lot of trouble when keying in programs using the alternate character set.

When using a CBM computer you may be tempted to select upper-case letters, using the SHIFT key, for the first letter of a word, or for the entire word. Consider the following statement:

```
10 For I=1 To 10 Step 2
```

Were this statement entered as illustrated in alternate character mode on a CBM computer, every upper-case letter would have to be generated by pressing the SHIFT key. Since the CBM computer assumes that the standard character set is present, this is how the BASIC statement illustrated above would be interpreted:

```
10 -OR ↘=1 ↓O 10 •TEP 2
```

The CBM computer would reject this statement and report a syntax error.

Within a printed text string shifted letters will not cause a syntax error.

OPERATING THE CASSETTE UNITS

Small CBM computer systems use cassette drives to store programs and data. Larger CBM computer systems use diskette drives for the same purpose. We will now give you step-by-step instructions to operate CBM computer cassette drives. Diskette drive operations are described next.

Before attempting to operate a cassette drive, make sure that the correct cassette tape is loaded in the drive. We are going to use drive 1, therefore load appropriate cassettes into drive 1, as described in Chapter 1. If the tape is not positioned at the beginning, press the REWIND button on the cassette unit; this will rewind the tape to its beginning. When rewound, press the STOP key. Make sure all of the cassette unit keys are in the off position (up).

The examples in this section will use the BLANKET program to illustrate the cassette operations. **Shaded portions of the examples designate your input; unshaded portions are the computer's response.**

Program SAVE

Saving a program on a cassette tape prevents it from being lost when the console's power is turned off. The SAVE statement writes the program which is currently in memory onto a cassette tape. Here is the SAVE statement format:

SAVE "program name"	*Save program on cassette unit # 1*
SAVE "program name",1	*Save program on cassette unit # 1*
SAVE "program name",2	*Save program on cassette unit # 2*

To save a program on cassette unit #1, use number 1 in the SAVE command. Use number 2 to save the program on cassette unit #2. When saving programs on cassette unit #1, the unit number is optional; you need not specify it. But, if you are saving a program on cassette unit #2, you must specify the unit number in the SAVE statement. In other words, #1 is the "default" cassette unit number, or the number selected when none is specified.

For practice, save the program BLANKET on a cassette tape. Load a blank cassette tape into cassette unit #1. Type in the program on the keyboard. To save the BLANKET program, enter the SAVE command and press the RETURN key. The computer responds by displaying the message PRESS PLAY & RECORD ON TAPE #*x*.

```
SAVE "BLANKET"

PRESS PLAY & RECORD ON TAPE #1
```

Simultaneously press the PLAY and RECORD buttons on the cassette unit. The cassette unit will move the tape forward as the BLANKET program is recorded onto the tape. Be careful to press both keys: if only the PLAY key is pressed, both you and the CBM computer will think that you are writing the program on the tape, but nothing will be written.

Once the PLAY and RECORD keys are pressed, the CBM computer responds with an OK and WRITING BLANKET message. When BLANKET has been successfully recorded the READY message and the cursor will appear on the screen. The entire screen display for a program save should look like this:

```
SAVE "BLANKET"

PRESS PLAY & RECORD ON TAPE #1
OK <------------------------- PLAY pressed on cassette unit #2

WRITING BLANKET
READY.
```

Caution: If *any* of the three tape movement keys REW, FFWD, or PLAY are depressed when the SAVE statement is executed, the CBM computer will start "writing" to the tape. Be sure to press the tape STOP key before issuing a SAVE statement; this assures that the CBM computer will print the PRESS PLAY & RECORD ON TAPE #*x* message. **Remember to depress the RECORD and PLAY keys at the same time.**

Following a program SAVE is a program VERIFY.

Program VERIFY

The VERIFY statement checks for recording errors in a saved program. By simulating the processes of a program LOAD, it reads and compares the program on the tape to the program in memory without actually loading the program into memory. If an error is detected, the CBM computer displays a warning message. Always verify a program after saving it.

To VERIFY a program type in:

VERIFY	*Verify next program encountered on tape cassette unit #2*
VERIFY "program name"	*Verify program on cassette unit #1*
VERIFY "program name",1	*Verify program on cassette unit #1*
VERIFY "program name",2	*Verify program on cassette unit #2*

Follow these steps to verify a program that you have just saved on cassette tape:

1. Rewind to the beginning of the tape, or to a point that you know precedes the beginning (load point) of the program just saved.
2. Press the tape STOP key.
3. Type in the VERIFY statement. You do not need to specify the program name, or cassette drive #1. You must specify cassette drive #2.

To verify the BLANKET program you just saved on the cassette unit, rewind the cassette tape to the beginning of the tape. Enter the VERIFY statement and press the RETURN key:

```
SAVE "BLANKET"

PRESS PLAY & RECORD ON TAPE #1
OK

WRITING BLANKET
READY.                          <-- Rewind and position tape before
VERIFY "BLANKET"                    start of BLANKET program

PRESS PLAY ON TAPE #1
OK                              <-- Press PLAY on cassette unit # 1

SEARCHING FOR BLANKET
FOUND BLANKET
VERIFYING
OK

READY.
```

When the PRESS PLAY ON TAPE #1 message appears on the screen, press the PLAY key on the cassette unit. The cassette unit will search the tape for the BLANKET program as the OK and SEARCHING FOR BLANKET comments are displayed on the screen. When found, the FOUND BLANKET message is displayed. The VERIFYING message appears on the screen as the BLANKET program on the tape is compared to the BLANKET program in the CBM computer memory. If the two compare with no errors, the screen displays an OK message, followed by a READY message and the cursor. If there are problems in the recording, the message ?VERIFY ERROR will appear. Rewind the tape and try to verify it again. If the error persists, rewind the tape back to the load point, save the program again, and verify the program. If the error still persists, try using another portion of the tape, or use another tape.

Program LOAD

The LOAD statement loads programs into computer memory. To load a program off cassette unit #1, type in either of the following two statements:

```
LOAD "program name"
LOAD "program name",1
```

When loading a program off of a cassette tape, the program name is optional. If the name is specified, the computer will load only the named program. If the program name is omitted, the computer will load the next program found on the cassette tape.

The optional number following the program name tells the computer which cassette unit to use. Cassette unit #1 (the built-in cassette unit on the PET 2001 or the external cassette unit attached in the rear interface on all other models) is assigned device number 1. The optional, second cassette unit is assigned device number 2.

If only cassette unit #1 is in use, you need not specify the device number; the LOAD statement automatically defaults to cassette unit #1. For example, if you want to load program BLANKET off a tape in cassette unit #1 you type in:

```
LOAD "BLANKET"

or: LOAD "BLANKET",1
```

Because cassette unit #1 is the default device, if you want to load a program off cassette unit #2 you must specify the #2, otherwise the computer will automatically search the tape in cassette unit #1.

The example below shows how the CBM computer responds to a LOAD statement. The shaded characters designate user input; unshaded characters are computer response:

```
LOAD "BLANKET"

PRESS PLAY ON TAPE #1 <---Press PLAY on cassette unit # 1
OK

SEARCHING FOR BLANKET
FOUND BLANKET
LOADING
READY.
```

As soon as the LOAD statement is input, the computer checks to see if the PLAY button is depressed and the tape is moving. If not, the message: PRESS PLAY ON TAPE #*x* is displayed. Once the PLAY button is pressed, the tape starts moving and the CRT responds with an OK message. The SEARCHING FOR message is displayed on the screen as the computer searches the tape for the program. Every time it finds a program, the word FOUND and the program name are displayed on the CRT. If the "found" program is not the correct program, the computer continues to search the tape, displaying the "found" program names until the correct program is found. Once found, LOADING is displayed as the program is loaded into computer memory. When the loading process is complete, the READY message is displayed.

After the first program has been loaded from a tape, you can for convenience leave the PLAY key depressed if you will be loading again from the tape. However, do not leave the PLAY key depressed if you will subsequently be saving a program. The computer can sense that a tape control key is depressed, but it cannot distinguish between them. A common error is to LOAD, then subsequently attempt to SAVE, with only the PLAY key depressed.

Sometimes you may have trouble loading a program from tape into memory. Often the CBM computer will find the program name but does not load the program correctly or does not load it at all. If this happens try to load the program again. Rewind the tape and reload (several times if necessary). If this does not work, there may have been a problem when saving the program (SAVE statement) or in the cassette unit itself.

Before issuing a LOAD statement, make sure the cassette tape is rewound to a point preceding the program you want to load. Otherwise the computer would search to the end of the tape without finding the program. If the program to be loaded is located at the end of the tape, and the computer searches from the beginning of the tape, the computer will find and load that program, but the operation will take a long time. If you want to save loading time, devise a scheme to locate approximately where the desired program is on the tape, and position the tape at a point just prior to the program. Then a LOAD statement will find the program quickly.

One scheme to locate information, rather than have the CBM computer search at its PLAY speed, is to use the measuring scale on the tape cassette; the one below goes from 0 to 100 in units of 10, with the beginning of the tape at 30:

As the tape moves forward, the tape radius, measured on the scale, goes from 30 up to 100. 100 is the end of the tape. By keeping a record of information stored on a tape, using the radius numbers, you can Fast Forward to a point just before the desired location.

After the program is loaded into computer memory, list it on the screen using the LIST statement, which was described earlier in this chapter.

Input the RUN statement in immediate mode to execute the program loaded off the cassette tape.

Program LOAD & RUN

"LOAD & RUN" automatically loads and executes the next program found on cassette tape #1. Pressing the shifted RUN/STOP key executes this operation, providing a quick way to load and run the next program stored on the cassette tape.

LOAD & RUN works only on CBM computer models with BASIC<3.0 (releases 1.x, 2.x, and 3.x).

To LOAD & RUN a program, position the cassette tape in cassette unit #1 before the beginning of the program to be loaded. Program names cannot be specified, therefore LOAD & RUN is usually used with cassette tapes that hold one program only. Press the shifted RUN key. The computer automatically loads and executes the next program found on the cassette tape, displaying all the regular LOAD messages. Here is an example of this method.

```
LΓ ←————————————————————————————Press shifted RUN/STOP key

PRESS PLAY ON TAPE #1←——————————————Press PLAY on cassette unit #1
OK

FOUND BLANKET
LOADING
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA←——Automatic program execution
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
.....
```

The shifted RUN key can only be pressed while the CBM computer is in immediate mode. If pressed during execution of another program, it acts like a STOP and breaks program execution.

OPERATING THE DISK UNIT

We will now explain how you load and execute programs that were previously stored on a diskette. We will also describe how you save new programs on a diskette.

No knowledge of BASIC programming is required to perform either of these tasks. Chapter 6 explains all disk operations and statements in detail. For now, just enter statements by rote.

By the end of this section you will know how to initialize both disk drives, list the diskette directory, and load, execute and save programs on diskette.

DOS RELEASES 1.x AND 2.x

A set of programs that come with your CBM computer control all disk operations. These programs are referred to collectively as a "Disk Operating System," or DOS.

There are currently several different releases of the CBM disk operating system. The first release, DOS 1.x (*x* is a number between 1 and 9 representing a subrelease of the DOS revision), is compatible with some CBM 2040 disk drive models and is used with BASIC<3.0. Major changes came with DOS release 2.x. Disk system statements were radically simplified, data storage and manipulation were improved, and the bugs in release 1.x were corrected. DOS 2.x works will all releases of BASIC. DOS 1.x works with BASIC<3.0. DOS 1.x will also work with BASIC 4.0 but some BASIC 4.0 disk commands will produce a disk syntax error.

All CBM 8050 disk drive models use the DOS release 2.x with BASIC 4.0. CBM 2040 model units use either DOS 1.x with BASIC<3.0 or 2.x with BASIC 4.0.

LOADING A PROGRAM FROM A DISKETTE USING BASIC <3.0

The following steps are involved when loading a program from a diskette:

1. **Open a logical file and device.**
2. **Initialize the disk drive(s).**
3. **List the directory. This step is optional. It lets you check on the exact spelling of the program name.**
4. **Load the program.**

To demonstrate the steps, we will use the TEST/DEMO diskette supplied by Commodore Business Machines with the disk unit.

Begin by inserting the TEST/DEMO diskette in drive 0 following the diskette loading procedure described in Chapter 1.

Opening a Logical File and Device

Before loading a program off a diskette, the communication line from the computer to the disk drive must be opened and readied for data transfer. The OPEN statement, input in immediate mode, opens the communication line.

To open a disk drive type in the following OPEN statement:

```
OPEN 1,8,15
```

Initializing the Disk Drive (BASIC<3.0)

The next step is to initialize the disk drive. **When using BASIC<3.0 the disk drive must be reinitialized every time a diskette is inserted in a drive, or when diskettes are switched between the two drives.** A disk drive must house a diskette before it is initialized.

You initialize the disk drive with the INITIALIZE statement. "I" is the abbreviation for INITIALIZE. Either may be used.

PRINT#file,# { Initialize / I } dr"

where:

file is the logical file number used in the OPEN command

dr is the disk drive number 0 or 1

To initialize drive 0 containing the TEST/DEMO diskette, type in:

```
PRINT#1,"I0"
```

You should see the drive 0 LED indicator light light up while the drive 0 motor runs briefly. When the motor stops and the drive indicator light goes out the drive is initialized.

To initialize drive 1, insert a diskette into drive 1 and type in:

```
PRINT#1,"I1"
```

To initialize both drives type in:

```
PRINT#1,"I"
```

Loading the Diskette Directory (DOS 1.x)

Step 3 loads and lists the disk directory into memory. This is an optional step; you can skip this step if you already know the name of the program you want to load.

The diskette directory serves the same function as the table of contents of a book: it names the programs stored on the diskette.

To load the directory off the diskette in drive 0 using BASIC<3.0, type in:

```
LOAD "$0",8
```

To load a directory off a diskette in drive 1 using BASIC< 3.0, type in:

```
LOAD "$1",8
```

If the drive number is not specified, both drives are searched and each diskette's directory is loaded.

```
LOAD "$",8
```

The following messages are displayed on the screen as the directory is loaded:

```
LOAD"$0",8

SEARCHING FOR $0
LOADING
READY.
```

When the READY message appears, type in LIST and press the RETURN key. LIST will list the directory contents on the screen; the cursor will disappear as the directory appears on the screen. If the directory is longer than 25 lines, initial lines will scroll off the top of the screen. To slow down the scrolling, hold down the REVERSE key or the ← key on the business keyboard while the directory is listing. To stop the list, press the STOP key.

You may list the directory as often as you want by retyping the LIST statement.

Here is an example of the TEST/DEMO diskette directory for a CBM 2040 model disk unit:

```
LIST

0 [redacted]
6     "DOS SUPPORT 4.0"   PRG
27    "DUM 3.4"           PRG
1     "  DISK DATA  "     SEQ
15    "DIAGNOSTIC BOOT"   PRG
10    "COPY DISK FILES"   PRG
4     "CHECK DISK"        PRG
10    "PET DISK"          PRG
10    "DISK DISPLAY"      PRG
3     "DISK COMM"         PRG
2     "DISK COMM2"        PRG
3     "DISK COMM3"        PRG
4     "DISK WRITE"        PRG
4     "DISK READ"         PRG
2     "DISK OVERLAYS"     PRG
5     "DISK DIR"          PRG
7     "  PET DATA  "      SEQ
34    "RANDOM 1.00"       PRG
27    "PRINTER DEMO"      PRG
12    "SEQUENTIAL 1.00"   PRG
484 BLOCKS FREE.
```

The top row of the directory is the "header." The header, displayed in reverse mode, shows the diskette name and identification number. The center column of the directory lists the program names. The quotation marks are not actually a part of the program name. The left column shows how many "blocks" of space on the diskette are filled by the program (a block is a unit of measure on the diskette). The number at the bottom of the column is the total number of unused blocks on the diskette. The rightmost column signals whether the information is saved as a program (PRG) or data. Do not worry about the left or right columns; they are of no importance at this time.

Once the directory is displayed scan the center column to find the program you wish to load.

Program LOAD (BASIC < 3.0)

The LOAD statement, input in immediate mode, loads the specified program from the diskette and stores it in computer memory.

To load any program from a diskette in drive 0, type in:

LOAD "0: program name ",8

The disk drive number precedes the program name; the two are separated by a colon. Together they are enclosed within quotation marks. If the disk drive number is not specified as 0 or 1, the disk operating system will search both drives for the program, providing both disk drives have previously been initialized.

To load the first program from the TEST/DEMO diskette in drive 0 using BASIC < 3.0, type in:

```
LOAD "0:DOS SUPPORT 4.0",8
```

Upon pressing RETURN the following messages appear on the screen, while the drive indicator light lights up and the disk drive makes a soft humming noise:

```
LOAD "0:DOS SUPPORT 4.0",8

SEARCHING FOR 0:DOS SUPPORT 4.0
LOADING
READY.
```

The disk drive number and program name are always displayed following the SEARCHING FOR message. When the LED light goes out and the humming noise stops, the READY message appears on the screen with the flashing cursor. The program is now loaded and ready to be listed and run.

Program LIST (BASIC <3.0)

Once a program is loaded into computer memory it may be listed on the display screen with the LIST statement, which we have already described. LIST displays the entire program. LIST with line parameters displays the specified portions of the program.

Program RUN (BASIC <3.0)

To execute the program just loaded off the diskette, type the RUN statement. Upon pressing the RETURN key, the program is executed.

Preparing a Blank Diskette for BASIC <3.0

A blank diskette cannot be used in the disk unit until it has been prepared, or "formatted." Following is the procedure to format a blank diskette, or to reformat an old diskette. Type in the statements by rote — you do not need to fully understand each step at this point.

OPEN the disk drive as follows:

```
OPEN 1,8,15
```

To format the diskette and intialize the disk drive the NEW statement is used. The format for the NEW statement is:

PRINT#file," { **N** / **NEW** } dr:diskname,id;"

where:

file	is the logical file number used in the OPEN command
dr	is the disk drive number 0 or 1
diskname	is the name assigned to the entire disk
id	is a unique two-character identifier.

Because your diskette is in drive 0, the disk drive number specified must be 0. Our example diskname is YAK, and the identification number is 01. Type in the following OPEN and NEW statements to format your diskette:

```
OPEN 1,8,15
PRINT#1,"N0:YAK"
```

Press the RETURN key. The drive 0 indicator light illuminates as the disk drive formats the diskette. The cursor disappears briefly, but then reappears with the READY message before the diskette format is complete; this allows you to do other things on the screen while the diskette is formatting. Once the operation is complete, type in:

```
LOAD "$0",8
```

$0 is the diskette directory for drive 0. $1 is the directory for drive 1. Directories display the diskette's "table of contents" on the screen. Each directory begins with a "header" with the diskname and identification number. When the directory is loaded, enter:

```
LIST
```

The directory will be displayed. Because the diskette is blank, only the header, shown here, is displayed:

```
LIST

 0 "WORK                 " 00 2A
 670 BLOCKS FREE.
READY.
```

Once the diskette is formatted you can store programs and data on it.

Program SAVE (BASIC < 3.0)

Recording a program on a diskette with the SAVE statement is similiar to recording onto a cassette tape. However, there are some differences in the syntax of the SAVE statement. A program SAVE for a diskette looks like this:

SAVE"dr:file name",**8**

where:

dr	is the disk drive number, 0 or 1
file	is the name assigned to the program

The number within the quotes and preceding the program name specifies the disk drive number, 0 or 1. It may not be omitted. The number following the program name, number 8, tells the computer to direct the program to the disk drive. If the number 8 is omitted the computer will try to save the program on the cassette unit O1.

Try saving the BLANKET program on a diskette in drive 0. Insert any formatted diskette into drive 0, then OPEN and INITIALIZE the drive, if not already done. Enter the BLANKET program on the keyboard. To save the program, enter the following SAVE statement and press the RETURN key.

```
OPEN 1,8,15 <-------- OPEN disk drive
PRINT#1,"I0"<-------- Initialize disk drive 0

10 FOR I=1 TO 800
20 PRINT "A"
30 NEXT I              Enter BLANKET program
40 PRINT "PHEW!"
50 END

SAVE "0:BLANKET",8 <-- Save BLANKET program on disk drive 0

READY.
```

As BLANKET is saved, the cursor will disappear from the screen and the indicator light on the disk drive will light up briefly. When the program has been completely recorded, the cursor and a READY message will appear on the screen. After saving a program, verify it.

Program VERIFY (BASIC<3.0)

Diskette and cassette programs are verified in the same way. The computer compares the saved program on the diskette against the program in memory. If an error is detected, an error message is displayed.

You should always VERIFY a program immediately after saving it.

The specifications for the VERIFY statement are identical to the SAVE statement. A program VERIFY for a diskette looks like this:

```
VERIFY"dr:filename",8
```

where:

dr	is the disk drive number, 0 or 2
filename	is the name of the program just saved

When the RETURN key is pressed, the verify messages (SEARCHING FOR BLANKET, VERIFYING, and OK) are displayed on the screen. If an error message appears instead of OK, then re-verify the program. If an error message appears again, resave the program, then re-verify.

```
SAVE "0:BLANKET",8 <-------- SAVE BLANKET program

READY.

VERIFY "0:BLANKET",8 <-------- Verify BLANKET program

SEARCHING FOR 0:BLANKET
VERIFYING <-------- Program verified
OK

READY.
```

Another way to VERIFY the program just saved on the diskette uses this statement:

```
VERIFY"*",8
```

The asterisk (*) signals the CBM computer to verify the program just saved on the diskette without having to specify the program name.

LOADING A PROGRAM FROM DISKETTE USING BASIC 4.0

There are two steps to loading a program from a diskette using BASIC 4.0:

1. **List the directory.**
2. **Load the program.**

To demonstrate these steps use any available prepared diskette or the DEMO diskette supplied by Commodore. The examples in this section will use the programs from the 8050 DEMO diskette.

Insert the diskette in drive 0 following the diskette loading procedure described in Chapter 1.

The BASIC 4.0 system automatically initializes the diskette drive before a program is loaded. However, **if you switch diskettes that have the same identification number** (the number following the diskette name on the directory header) then **you should manually initialize the drive**, otherwise the computer will not know that the diskettes have been switched. Drives are initialized when you load the directory.

Loading the Diskette Directory (BASIC 4.0)

The DIRECTORY statement loads the diskette directory and displays it on the screen.

To load the diskette directory type in:

DIRECTORY D0	*Load and list the directory of disk drive 0*
DIRECTORY D1	*Load and list the directory of disk drive 1*
DIRECTORY	*Load and list the directory of disk drives 0 and 1*

If the disk drive is not specified, both directories are loaded and listed. A directory may *not* be listed or run like a regular CBM BASIC program under BASIC 4.0.

A sample diskette directory in drive 0 appears as follows:

```
DIRECTORY D0

 0 [redacted]
 5     "UNIVERSAL WEDGE"   PRG
 8     "UNIT TO UNIT"      PRG
 3     "CHANGE 8050"       PRG
 11    "COPY 2040 - 8050"  PRG
 27    "PRINTER DEMO"      PRG
 12    "SEQUENTIAL"        PRG
 11    "PERFORMANCE TEST"  PRG
 5     "CHECK DISK"        PRG
 17    "LOGIC DIAGNOSTIC"  PRG
 1953 BLOCKS FREE.
READY.
```

Program LOAD (BASIC 4.0)

Programs are loaded from a diskette with the DLOAD statement in immediate mode. To load a program from a diskette type in:

DLOAD "program name" **DLOAD** "program name",**D0**	*Load program from disk drive 0*
DLOAD "program name",**D1**	*Load program from disk drive 0*

The program name is mandatory and must be enclosed within quotation marks. If the disk drive is not specified the drive number defaults to 0 and only drive 0 is searched. If a program is not found, a ?FILE NOT FOUND error results.

Try these statements on your disk system. Load the directory of the DEMO diskette or your own diskette in drive 0. Type in:

```
DIRECTORY D0
```

The cursor should disappear briefly while the diskette is initialized and the directory is loaded into memory and displayed.

Next, load the second program listed on the directory. (In the following example, the program name may differ from your program name — don't worry about it.) Type in:

```
DLOAD "UNIT TO UNIT"
```

The normal load messages appear on the screen as the diskette is searched and the UNIT TO UNIT program is loaded into memory.

```
DLOAD "UNIT TO UNIT",D0

SEARCHING FOR UNIT TO UNIT
LOADING
READY.
```

Program LIST (BASIC 4.0)

To list the program, use the LIST statement, which we described earlier.

The LIST statement is used in the same way by BASIC 4.0 and BASIC<3.0. The LIST statement description given earlier applies to all DOS releases.

Program RUN (BASIC 4.0)

To execute the program, type in the RUN statement:

```
RUN
```

Program LOAD & RUN (BASIC 4.0)

The LOAD & RUN function automatically loads and executes the first program found on disk drive 0. Pressing the shifted RUN/STOP key executes this operation, providing a quick way to load and run the first program on the drive 0 diskette.

The disk drive LOAD & RUN works only on CBM models with BASIC 4.0 (release 4.x).

To LOAD & RUN a program, simply press the shifted RUN key. The screen displays the following messages before the program begins execution:

```
                    <-------------- Shifted RUN key pressed
DL"*

SEARCHING FOR 0:*
LOADING
                    <-------------- Program execution begins
```

To perform a LOAD & RUN, the shifted RUN key can only be pressed while the CBM computer is in immediate mode. If pressed during execution of another program, it acts like a STOP and breaks program execution.

Preparing a Blank Diskette for BASIC 4.0

A blank diskette cannot be used in the disk unit until it has been prepared, or "formatted." **Following is the procedure to format a blank diskette, or to reformat an old diskette.** Type in the commands by rote — you do not need to fully understand each step at this point.

Insert a diskette in drive 0.

Diskettes are formatted with the HEADER statement:

HEADER "diskname",Dx,I22

where:

diskname	is the name to be assigned to the disk
x	is the disk drive number, 0 or 1
22	is the two-character identification number

Type in the following HEADER statement to format your diskette:

```
HEADER "YAK", D0, I01
```

Our example disk name is YAK. Because the diskette is in drive 0, the disk drive number specified must be D0. If the diskette is in drive 1, then D1 is specified. The identification number we will use is 1, although any unique two-letter combination of letters or numbers may be used.

When you press the RETURN key, the computer will respond with an ARE YOU SURE? message. Press the 'Y' key for yes, or the 'N' key for no. Press the RETURN key. 'N' will cause an exit from the format procedure. If 'Y' is pressed, the drive 0 indicator light illuminates as the disk drive formats the diskette. The cursor disappears for a short time, reappearing with a READY message when the format operation is complete.

```
HEADER "YAK", D0, I01
ARE YOU SURE ?Y ←———— Y key pressed for "yes"
READY.
■
```

Once the diskette is formatted you can write program and data files on it.

Program SAVE (BASIC 4.0)

Programs are saved on a diskette using the DSAVE statement in immediate mode. To save a program on a diskette type in:

DSAVE "filename",Dx

where:

filename	is the name of the program
x	is the disk drive number 0 or 1

The program name is mandatory and must be enclosed within quotation marks. If the disk drive number is not specified, the program is automatically saved on the diskette in drive 0.

Try this statement by saving the BLANKET program on a formatted diskette in drive 0. Enter the BLANKET program on the keyboard. Then type in the following DSAVE statement:

```
DSAVE "BLANKET",D0
```

Upon pressing the RETURN key, the indicator light on the disk drive will light up briefly and the cursor will disappear from the screen. Upon completion of the save, a READY message and the cursor will return to the screen as shown:

```
DSAVE "BLANKET",D0

READY.
■
```

The BLANKET program should now be verified.

Program VERIFY (BASIC 4.0)

The BASIC 4.0 and BASIC<3.0 VERIFY statements are the same. Refer to the VERIFY statement under BASIC<3.0 for the description.

OPERATING THE CBM PRINTER

The CBM printer can be used to list program statements and to print results. Like disk and tape units, the printer can be controlled by statements entered in immediate mode via the keyboard, or using statements within a program. We are going to describe immediate mode printer control. Chapter 6 explains how to control a printer using program statements. **Printer operations are identical using BASIC 4.0 or BASIC<3.0.**

The OPEN Statement

Prior to sending data to the printer, you must open a communication line from the computer to the printer. This is done using an OPEN statement, as follows:

```
OPEN x,4
```

where:

x is any integer from 1 to 255

Here are some examples of OPEN statements:

```
OPEN 1,4

OPEN 4,4

OPEN 250,4
```

The CMD Statement

Once the printer has been opened, output is directed to the printer using the CMD statement. This is optional.

```
CMD 1
```

The CMD number must match the first number of the OPEN statement. If the numbers do not match, the message ?FILE NOT OPEN ERROR will be displayed. If this occurs, the printer will have to be reopened by retyping the OPEN and CMD statements.

Here are some examples with PRINT statements:

```
OPEN 1,4 CMD 4
PRINT "TISHNICK"


OPEN 2,4
CMD 2: "MY PET BITES"


OPEN 3,4
PRINT#3, "54321"
```

Printing to the Printer

Once the printer has been properly opened, it is ready to receive and print data. We will use the PRINT# statement, entered in immediate mode, to print data at the printer.

First make sure the printer is connected to the computer and powered up, with ribbon and paper properly installed.

Open the printer by typing in OPEN and CMD statements:

```
OPEN 1,4:CMD 1
```

Upon pressing RETURN, the printer executes a line feed (prints one blank line). The printer is now open and ready to print data input from the keyboard. Type in the following PRINT statement, placing your name between the quotation marks:

```
PRINT "KIT CARSON"
```

When you press RETURN, the cursor will disappear from the screen as your name is printed at the printer:

```
READY.
KIT CARSON
```

When printing in immediate mode, the first line printed is a READY message. The following line is your output; in this case, your name. At the end of the output the printer automatically executes a carriage return and another line feed. When the cursor reappears on the screen, you can input more data.

Below are some more sample inputs and printer outputs.

Screen Display **Printer Output**

```
OPEN 1,4:CMD 1
PRINT "KIT CARSON"
PRINT "1234567890"
PRINT "MY NAME BACKWARDS IS TIK NOSRAC"
```

```
KIT CARSON

READY.
1234567890

READY.
MY NAME BACKWARDS IS TIK NO RAC
```

Listing a Program on the Printer

To list a program on the printer, type in the OPEN and CMD statements, followed by the LIST statement.

```
OPEN 1,4:CMD 1
LIST
```

LIST line parameters may also be used.

The example below shows the BLANKET program listed by the printer:

```
OPEN 1,4:CMD 1
LIST

 10 FOR I=1 TO 800
 20 PRINT "A";
 30 NEXT I
 40 PRINT "PHEW!"
 50 END
READY.
```

The CLOSE Statement

You must CLOSE the printer after using it. The printer is closed with a CLOSE statement, as follows:

```
CLOSE 1
```

The number following CLOSE must be the first number in the OPEN statement.

```
OPEN 1, 4                    OPEN 15, 4
.                            .
.                            .
.                            .
CLOSE 1                      CLOSE 15
```

You must precede the CLOSE statement with a PRINT# statement to properly close the printer. Below are examples of correct and incorrect ways to close the printer:

Right		Wrong
OPEN 5, 4 PRINT#5,"HELLO THERE" CLOSE 5		
OPEN 5, 4 CMD 5,"HELLO THERE" PRINT#5:CLOSE 5	not	OPEN 5, 4 CMD 5, "HELLO THERE" CLOSE 5
OPEN 5,4 CMD 5,"HELLO THERE" PRINT#5,"HELLO THERE" CLOSE 5	not	OPEN 5,4 CMD 5,"HELLO THERE" PRINT#5,"HELLO THERE" PRINT#5:CLOSE 5
OPEN 5, 4 PRINT#5,"HELLO THERE" CMD5,"HELLO THERE" PRINT#5:CLOSE 5	not	OPEN 5, 4 PRINT#5,"HELLO THERE" CMD5,"HELLO THERE" CLOSE 5