Machine Language	Assembly Language
01010101	PUSH BP
10001011 11101100	MOV BP,SP
01001100	DEC SP
01001100	DEC SP
01010110	PUSH SI
01010111	PUSH DI
10111111 00000011 00000000	MOV DI, 0003
10111110 00000010 00000000	MOV SI, 0002
10001011 11000111	MOV AX, DI
00000011 11000110	ADD AX, SI
10001001 01000110 11111110	MOV [BP-02], AX
01011111	POP DI
01011110	POP SI
10001011 11100101	MOV SP, BP
01011110	POP BP
11000011	RET

FIGURE 2-8
In assembly language, each microprocessor instruction is assigned a code that makes the program more meaningful to people. It is still difficult, however, for the untrained person to see what the program will do.

set than the microprocessors in DOS-compatible computers. An assembly language program written for a DOS computer will not work on a Macintosh. However, a simple program written in a high-level language can work on both computers with little or no modification.

So why use a low-level language? It depends on what you need to do. The drawback of high-level languages is that they do not always provide a command for everything the programmer wants a program to do. Using assembly language, the programmer can write instructions that enable the computer to do anything the hardware will allow.

Another advantage of low-level languages is that a program written in a low-level language will generally require less memory and run more quickly than the same program written in a high-level language. This is because high-level languages must be translated into machine language before the microprocessor can execute the instructions. The translation is done by another

BASIC	Pascal	C++
10 I = 3	program AddIt;	main()
20 J = 2		{
30 K = I + J	var	int i,j,k;
	i, j, k : integer;	i = 3;
		j = 2;
	begin	k = i + j;
	i := 3;	return 0;
	j := 2;	}
	k := i + j;	
	end.	

FIGURE 2-9
The same program can be written in more than one high-level language.

program, and is usually less efficient than the work of a skilled assembly-language programmer. Table 2-2 summarizes the advantages of low- and high-level languages.

TABLE 2-2

ADVANTAGES OF LOW-LEVEL LANGUAGES	ADVANTAGES OF HIGH-LEVEL LANGUAGES
Better use of hardware's capabilities	Requires less programming
Requires less memory	Fewer programming errors
Runs more quickly	Easier to move among computers with different microprocessors
	More easily read

INTERPRETERS AND COMPILERS

Programmers writing in a high-level language enter the program's instructions into a text editor. A ***text editor*** is similar to a word processor, except the files are saved in ASCII format without the font and formatting codes word processors use. The files saved by text editors are called ***text files.*** A program in the form of a high-level language is called ***source code.***

Programmers must have their high-level programs translated into the machine language the microprocessor understands. The translation may be done by interpreters or compilers. The resulting machine language code is known as ***object code.***

INTERPRETERS

An ***interpreter*** is a program that translates the source code of a high-level language into machine language. An interpreter translates a computer language in a way similar to the way a person might interpret between languages like English and Spanish. Each instruction is interpreted from the programming language into machine language as the instructions are needed. Interpreters are normally used only with very high-level languages, such as most versions of BASIC.

To run a program written in an interpreted language, you must first load the interpreter into the computer's memory. Then you load the program to be interpreted. The interpreter steps through the program one instruction at a time and translates the instruction into machine language, which is sent to the microprocessor. Every time the program is run, the interpreter must once again translate each instruction.

Because of the need to have the interpreter in memory before the program can be interpreted, interpreted languages are not widely used to write programs that are sold. The buyer of the program would have to have the correct interpreter in order to use the program.

COMPILERS

A ***compiler*** is another program that translates a high-level language into machine language. A compiler, however, makes the translation once, then saves the machine language so that the instructions do not have to be translated each time