During the maintenance phase of the programming process, bugs may be found that were not uncovered during the testing and debugging phase. It is also possible that better ways to accomplish a task in the program may be discovered. It is important to understand that the steps of the programming process may be repeated in order to refine or repair an existing program.

SECTION 2.3 QUESTIONS

1. List the five basic steps in the programming process.
2. Define algorithm.
3. Give an example of an algorithm used in everyday life.
4. Define the term *bug* as it relates to programming.
5. What is the purpose of documentation inside a program?
6. Write an algorithm that gives directions from one location to another. Choose a starting point (your home, for example), and give detailed, step-by-step directions that will lead anyone who might be reading the algorithm to the correct destination.
7. Draw a flowchart that describes the steps you follow when you get up in the morning and get ready for your day. Include as many details as you want, including things such as hitting the snooze button on your alarm clock, brushing your teeth, and eating breakfast.

KEY TERMS

algorithm
American Standard Code for Information Interchange (ASCII)
analog
assembler
assembly language
binary number system
bit
bugs
byte
C++
characters
compiler
crash
data
decimal number system
digital
executable file
flowchart
graphical user interface (GUI)
high-level language
interpreter
linker
low-level language
machine language
object code
object file
operating system
programming language
pseudocode
run-time error
source code
states
syntax error
text editor
text file

SUMMARY

- Inside a computer, signals called bits represent data and give instructions. Bits are commonly arranged in groups of eight, called bytes. At the heart of the work a computer does is a device called a microprocessor. The microprocessor responds to commands called machine language.
- High-level programming languages allow programmers to work in a language that people can more easily read. Machine language and assembly language are low-level languages because each instruction in the language corresponds to one or only a few microprocessor instructions. In high-level languages, instructions may represent many microprocessor instructions.
- High-level languages must be translated into machine language by an interpreter or compiler. An interpreter translates each program step into machine language as the program runs. A compiler translates the program before it is run, and saves the machine language as an object file. A linker then creates an executable file from the object file.
- Input and output operations and loading of executable files are handled by the operating system. The operating system loads a program and turns over control of the system to the program. When the program ends, the operating system takes control again.
- Programming involves five basic steps: defining the problem; developing an algorithm; coding the program; testing and debugging; and documenting and maintaining.

PROJECTS

PROJECT 2-1

Choose some large value, like the salary of your favorite professional athlete, and convert it to the binary number system.

PROJECT 2-2

Make a chart of at least 12 high-level languages. Include a brief description of each language that tells the primary use of the language or its historical significance. If you can find the date the language was created, include that on your chart. Some languages to consider are Ada, ALGOL, BASIC, C, C++, COBOL, FORTRAN, LISP, Logo, Oberon, Pascal, PL/I, and Smalltalk.

PROJECT 2-3

Choose a computer program with which you are familiar. List the inputs and outputs of the program. Draw a basic flowchart for the program. If the program is too large for a simple flowchart, draw a flowchart for one part of the program.