- A while loop executes one or more statements as long as the control expression is true. The control expression is tested before the statements in the loop begin. A do while loop works like a while loop except the control expression is tested at the end of the loop.
- A break statement can be used to exit a loop before the control expression ends the loop. The continue statement causes the loop to skip to the next iteration of the loop.
- A loop within a loop is called a nested loop. The more deeply nested the loop, the more times the loop will be executed.

PROJECTS

PROJECT 8-1

1. Draw a flow chart for a simple program of your own design that uses a while loop.
2. Write the C++ source code for the program.
3. Enter the source code into a blank editor screen and give it an appropriate filename.
4. Compile and run the program. Close.

PROJECT 8-2

Write a program that uses nested loops to produce the following output.

```
A1B1B2B3A2B1B2B3
```

PROJECT 8-3 • FINANCIAL PLANNING

Write a program that calculates the amount of time necessary to reach a certain financial goal by consistently depositing the same amount of money into an interest-bearing account each month. The account is compounded monthly.

PROJECT 8-4 • GAME PROGRAMMING

Write a program that asks the user to think of a number between 1 and 100, then attempts to guess the number. The program should make an initial guess of 50. The program should then ask the user if 50 is the number the user has in mind, or if 50 is too high or too low. Based on the response given by the user, the program should make another guess. Your program must continue to guess until the correct number is reached. Save the source file as *HI-LO.CPP*.

PROJECT 8-5 • NUMBER SYSTEMS

1. Open *BINARY.CPP*. The program uses four nested loops to print the binary equivalent of 0 to 15 to the screen. Study the source code and run the program to see its output.
2. Modify the program to generate an additional column of digits. The resulting output should be the binary equivalent of 0-31. Save the modified source file as *BINARY31.CPP*.
3. Close the source code file.

PROJECT 8-6 • NUMBER SYSTEMS

Write a program that converts a binary number (up to seven digits) to a decimal value.

PROJECT 8-7 • NUMBER SYSTEMS

Write a program that converts standard Arabic numbers to Roman numerals.

PROJECT 8-8 • NUMBER SYSTEMS

Extend the program from Project 8-7 to convert from Roman numerals to standard Arabic numbers.

PROJECT 8-9 • MAKING CHANGE

Write a program that calculates the number of quarters, dimes, nickels, and pennies necessary to generate the number of cents entered as input. For example, if 93 cents is entered as input, the program should indicate that three quarters, one dime, one nickel, and three pennies are necessary to add up to 93 cents.

PROJECT 8-10 • MATHEMATICS

Write a program that finds the integer from 1 to 1000 with the most divisors that produce no remainder. For example, the integer 60 has 12 divisors that produce no remainder. They are 1, 2, 3, 4, 5, 6, 10, 12, 15, 20, 30, 60.

PROJECT 8-11 • MATHEMATICS

Write a program that will reduce fractions. Ask the user for the numerator and the denominator. Output a new, reduced fraction, and the greatest common factor.