EX 5.14 Write a method called `complexEquation` that accepts three integer parameters and returns true, if the first parameter is not equal to the second and the product of the first two parameters is equal to the third, and false, otherwise.

EX 5.15 Write a method called `isAlpha` that accepts a character parameter and returns true if that character is either an uppercase or lowercase alphabetic letter.

EX 5.16 Write a method called `floatEquals` that accepts three floating point values as parameters. The method should return true if the first two parameters are equal within the tolerance of the third parameter.

EX 5.17 Write a method called `isIsosceles` that accepts three integer parameters that represent the lengths of the sides of a triangle. The method returns true if the triangle is isosceles but not equilateral (meaning that exactly two of the sides have an equal length), and false otherwise.

EX 5.18 Explain what would happen if the radio buttons used in the `QuoteOptions` program were not organized into a `ButtonGroup` object. Modify the program to test your answer.

**Programming Projects**

PP 5.1 Write a program that reads an integer value from the user representing a year. The purpose of the program is to determine if the year is a leap year (and therefore has 29 days in February) in the Gregorian calendar. A year is a leap year if it is divisible by 4, unless it is also divisible by 100 but not 400. For example, the year 2003 is not a leap year, but 2004 is. The year 1900 is not a leap year because it is divisible by 100, but the year 2000 is a leap year because even though it is divisible by 100, it is also divisible by 400. Produce an error message for any input value less than 1582 (the year the Gregorian calendar was adopted).

PP 5.2 Modify the solution to the previous project so that the user can evaluate multiple years. Allow the user to terminate the program using an appropriate sentinel value. Validate each input value to ensure it is greater than or equal to 1582.

PP 5.3 Write a program that calculates and prints, the sum and product of all the digits in an integer value read from the keyboard.
PP 5.4 Write a program that plays the Hi-Lo guessing game with numbers. The program should pick a random number between 1 and 100 (inclusive), then repeatedly prompt the user to guess the number. On each guess, report to the user that he or she is correct or that the guess is high or low. Continue accepting guesses until the user guesses correctly or chooses to quit. Use a sentinel value to determine whether the user wants to quit. Count the number of guesses and report that value when the user guesses correctly. At the end of each game (by quitting or a correct guess), prompt to determine whether the user wants to play again. Continue playing games until the user chooses to stop.

PP 5.5 Create a ChangeCase program which considers a string as an input from the user and identifies whether the characters of the string are in uppercase or lowercase, then changes all the lowercase characters, if any, to uppercase and vice versa.

PP 5.6 Design a Person class similar to the Coin class defined in this chapter. Then design and implement a driver class called SelectPerson whose main method creates two Person objects, then randomly selects a gender in both to see in which object a female gender is selected two times in a row. Consider the possibility that they might tie. Print the results of each selection and then print the object name which comes up first with the female gender being selected twice, consecutively.

PP 5.7 Write a program that plays the Rock-Paper-Scissors game against the computer. When played between two people, each person picks one of three options (usually shown by a hand gesture) at the same time, and a winner is determined. In the game, Rock beats Scissors, Scissors beats Paper, and Paper beats Rock. The program should randomly choose one of the three options (without revealing it), then prompt for the user’s selection. At that point, the program reveals both choices and prints a statement indicating if the user won, the computer won, or if it was a tie. Continue playing until the user chooses to stop, then print the number of user wins, losses, and ties.

PP 5.8 Design and implement an application that simulates an odd slot machine in which five numbers between 1 and 100 are randomly selected and printed side by side. Print an appropriate statement if all five of the numbers are odd, or if any one of the numbers is odd. Continue simulating until the user chooses to stop.