Program 5-6 *(continued)*

```
11       int num = MIN_NUMBER;     // Counter
12
13       cout << "Number Number Squared\n";
14       cout << "-------------------------\n";
15       while (num <= MAX_NUMBER)
16       {
17            cout << num << "\t\t" << (num * num) << endl;
18            num++; //Increment the counter.
19       }
20       return 0;
21  }
```

Program Output

```
Number Number Squared
---------------------
1            1
2            4
3            9
4            16
5            25
6            36
7            49
8            64
9            81
10           100
```

In Program 5-6, the variable num, which starts at 1, is incremented each time through the loop. When num reaches 11 the loop stops. num is used as a *counter* variable, which means it is regularly incremented in each iteration of the loop. In essence, num keeps count of the number of iterations the loop has performed.

NOTE: It's important that num be properly initialized. Remember, variables defined inside a function have no guaranteed starting value.

5.5 The do-while Loop

CONCEPT: The **do-while** loop is a posttest loop, which means its expression is tested after each iteration.

The do-while loop looks something like an inverted while loop. Here is the do-while loop's format when the body of the loop contains only a single statement:

```
do
  statement;
while (expression);
```

Here is the format of the do-while loop when the body of the loop contains multiple statements:

```
do
{
  statement;
  statement;
  // Place as many statements here
  // as necessary.
} while (expression);
```

NOTE: The do-while loop must be terminated with a semicolon.

The do-while loop is a *posttest* loop. This means it does not test its expression until it has completed an iteration. As a result, the do-while loop always performs at least one iteration, even if the expression is false to begin with. This differs from the behavior of a while loop, which you will recall is a pretest loop. For example, in the following while loop the cout statement will not execute at all:

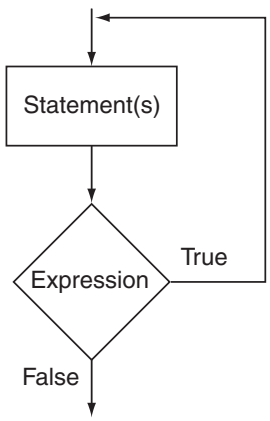
```
int x = 1;
while (x < 0)
   cout << x << endl;
```

But the cout statement in the following do-while loop will execute once because the do-while loop does not evaluate the expression x < 0 until the end of the iteration.

```
int x = 1;
do
   cout << x << endl;
while (x < 0);
```

Figure 5-5 illustrates the logic of the do-while loop.

Figure 5-5



You should use the do-while loop when you want to make sure the loop executes at least once. For example, Program 5-7 averages a series of three test scores for a student. After the

average is displayed, it asks the user if he or she wants to average another set of test scores. The program repeats as long as the user enters Y for yes.

Program 5-7

```
1  // This program averages 3 test scores. It repeats as
2  // many times as the user wishes.
3  #include <iostream>
4  using namespace std;
5
6  int main()
7  {
8       int score1, score2, score3; // Three scores
9       double average;             // Average score
10      char again;                 // To hold Y or N input
11
12      do
13      {
14           // Get three scores.
15           cout << "Enter 3 scores and I will average them: ";
16           cin >> score1 >> score2 >> score3;
17
18           // Calculate and display the average.
19           average = (score1 + score2 + score3) / 3.0;
20           cout << "The average is " << average << ".\n";
21
22           // Does the user want to average another set?
23           cout << "Do you want to average another set? (Y/N) ";
24           cin >> again;
25      } while (again == 'Y' || again == 'y');
26      return 0;
27 }
```

Program Output with Example Input Shown in Bold

```
Enter 3 scores and I will average them: 80 90 70 [Enter]
The average is 80.
Do you want to average another set? (Y/N) y [Enter]
Enter 3 scores and I will average them: 60 75 88 [Enter]
The average is 74.3333.
Do you want to average another set? (Y/N) n [Enter]
```

When this program was written, the programmer had no way of knowing the number of times the loop would iterate. This is because the loop asks the user if he or she wants to repeat the process. This type of loop is known as a *user-controlled loop*, because it allows the user to decide the number of iterations.

Using `do-while` with Menus

The `do-while` loop is a good choice for repeating a menu. Recall Program 4-27, which displayed a menu of health club packages. Program 5-8 is a modification of that program, which uses a `do-while` loop to repeat the program until the user selects item 4 from the menu.

Program 5-8

```
// This program displays a menu and asks the user to make a
// selection. A do-while loop repeats the program until the
// user selects item 4 from the menu.
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    // Constants for menu choices
    const int ADULT_CHOICE = 1,
              CHILD_CHOICE = 2,
              SENIOR_CHOICE = 3,
              QUIT_CHOICE = 4;

    // Constants for membership rates
    const double ADULT = 40.0,
                 CHILD = 20.0,
                 SENIOR = 30.0;

    // Variables
    int choice;        // Menu choice
    int months;        // Number of months
    double charges;    // Monthly charges

    // Set up numeric output formatting.
    cout << fixed << showpoint << setprecision(2);

    do
    {
        // Display the menu.
        cout << "\n\t\tHealth Club Membership Menu\n\n"
             << "1. Standard Adult Membership\n"
             << "2. Child Membership\n"
             << "3. Senior Citizen Membership\n"
             << "4. Quit the Program\n\n"
             << "Enter your choice: ";
        cin >> choice;

        // Validate the menu selection.
        while (choice < ADULT_CHOICE || choice > QUIT_CHOICE)
        {
            cout << "Please enter a valid menu choice: ";
            cin >> choice;
        }

        // Process the user's choice.
        if (choice != QUIT_CHOICE)
        {
            // Get the number of months.
            cout << "For how many months? ";
```

(program continues)

Program 5-8 *(continued)*

```
52                cin >> months;
53
54                // Respond to the user's menu selection.
55                switch (choice)
56                {
57                     case ADULT_CHOICE:
58                          charges = months * ADULT;
59                          break;
60                     case CHILD_CHOICE:
61                          charges = months * CHILD;
62                          break;
63                     case SENIOR_CHOICE:
64                          charges = months * SENIOR;
65                }
66
67                // Display the monthly charges.
68                cout << "The total charges are $"
69                     << charges << endl;
70           }
71       } while (choice != QUIT_CHOICE);
72       return 0;
73  }
```

Program Output with Example Input Shown in Bold

```
                Health Club Membership Menu

1. Standard Adult Membership
2. Child Membership
3. Senior Citizen Membership
4. Quit the Program

Enter your choice: 1 [Enter]
For how many months? 12 [Enter]
The total charges are $480.00

                Health Club Membership Menu

1. Standard Adult Membership
2. Child Membership
3. Senior Citizen Membership
4. Quit the Program

Enter your choice: 4 [Enter]
Program ending.
```

Checkpoint

5.5 What will the following program segments display?

A)
```
int count = 10;
do
{
   cout << "Hello World\n";
   count++;
} while (count < 1);
```