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

p163

Program 4-6

```
 1 // This program averages 3 test scores.
 2 // It demonstrates an if statement executing
 3 // a block of statements.
 4 #include <iostream>
 5 #include <iomanip>
 6 using namespace std;
 7
 8 int main()
 9 {
10    const int HIGH_SCORE = 95;   // A high score is 95 or greater
11    int score1, score2, score3;  // To hold three test scores
12    double average;              // TO hold the average score
13
14    // Get the three test scores.
15    cout << "Enter 3 test scores and I will average them: ";
16    cin >> score1 >> score2 >> score3;
17
18    // Calculate and display the average score.
19    average = (score1 + score2 + score3) / 3.0;
20    cout << fixed << showpoint << setprecision(1);
21    cout << "Your average is " << average << endl;
22
23    // If the average is high, congratulate the user.
24    if (average > HIGH_SCORE)
25    {
26       cout << "Congratulations!\n";
27       cout << "That's a high score.\n";
28       cout << "You deserve a pat on the back!\n";
29    }
30    return 0;
31 }
```

p171

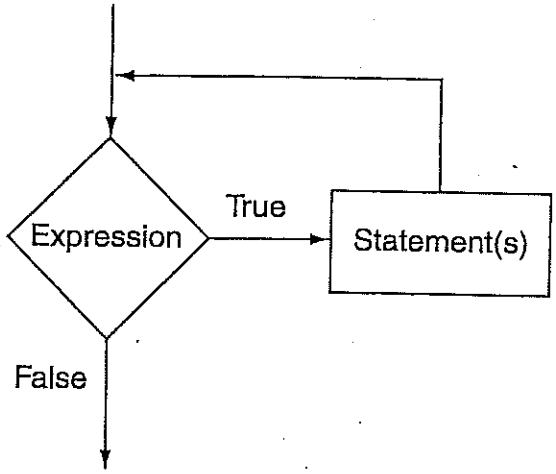
Program 4-11

```
1  // This program demonstrates the nested if statement.
2  #include <iostream>
3  using namespace std;
4
5  int main()
6  {
7     char employed,   // Currently employed, Y or N
8          recentGrad; // Recent graduate, Y or N
9
10    // Is the user employed and a recent graduate?
11    cout << "Answer the following questions\n";
12    cout << "with either Y for Yes or ";
13    cout << "N for No.\n";
14    cout << "Are you employed? ";
15    cin >> employed;
16    cout << "Have you graduated from college ";
17    cout << "in the past two years? ";
18    cin >> recentGrad;
19
20    // Determine the user's loan qualifications.
21    if (employed == 'Y')
22    {
23       if (recentGrad == 'Y') // Nested if
24       {
25          cout << "You qualify for the special ";
26          cout << "interest rate.\n";
27       }
28       else // Not a recent grad, but employed
29       {
30          cout << "You must have graduated from ";
31          cout << "college in the past two\n";
32          cout << "years to qualify.\n";
33       }
34    }
35    else // Not employed
36    {
37       cout << "You must be employed to qualify.\n";
38    }
39    return 0;
40 }
```

The `while` Loop

The `while` loop has two important parts: (1) an expression that is tested for a true or false value, and (2) a statement or block that is repeated as long as the expression is true. Figure 5-1 shows the logic of a `while` loop.

Figure 5-1



p233

Here is the general format of the while loop:

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

In the general format, *`expression`* is any expression that can be evaluated as true or false, and *`statement`* is any valid C++ statement. The first line shown in the format is sometimes called the *loop header*. It consists of the key word `while` followed by an *`expression`* enclosed in parentheses.

Here's how the loop works: the *`expression`* is tested, and if it is true, the *`statement`* is executed. Then, the *`expression`* is tested again. If it is true, the *`statement`* is executed. This cycle repeats until the *`expression`* is false.

The statement that is repeated is known as the *body* of the loop. It is also considered a conditionally executed statement, because it is executed only under the condition that the *`expression`* is true.

Notice there is no semicolon after the expression in parentheses. Like the `if` statement, the `while` loop is not complete without the statement that follows it.

If you wish the `while` loop to repeat a block of statements, its format is:

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

Program 5-3

```
1  // This program demonstrates a simple while loop.
2  #include <iostream>
3  using namespace std;
4
5  int main()
6  {
7     int number = 0;
8
9     while (number < 5)
10    {
11       cout << "Hello\n";
12       number++;
13    }
14    cout << "That's all!\n";
15    return 0;
16 }
```

p234

Program Output

```
Hello
Hello
Hello
Hello
Hello
That's all!
```

p236

Don't Forget the Braces with a Block of Statements

If you write a loop that conditionally executes a block of statements, don't forget to enclose all of the statements in a set of braces. If the braces are accidentally left out, the `while` statement conditionally executes only the very next statement. For example, look at the following code.

```
int number = 0;
// This loop is missing its braces!
while (number < 5)
   cout << "Hello\n";
   number++;
```

{ }

In this code the `number++` statement is not in the body of the loop. Because the braces are missing, the `while` statement only executes the statement that immediately follows it. This loop will execute infinitely because there is no code in its body that changes the `number` variable.