little difficulty using *if* in C++. The if structure is one of the easiest and most useful parts of C++.

The expression that makes the decision is called the ***control expression.*** Look at the code segment below. First the control expression (i == 3) is evaluated. If the result is true, the code in the braces that follow the *if* is executed. If the result is false, the code in the braces is skipped.

```
if (i == 3)
  { cout << "The value of i is 3\n"; }
```

Note

When only one statement appears between the braces in an if structure, the braces are not actually necessary. It is, however, a good idea to always use braces in case other statements are added later.

PITFALLS

Remember to be careful of confusing the == operator with the = (assignment) operator. Usage like `if (i = 3)` *will cause i to be assigned the value 3 and the code in the braces that follow will be executed regardless of what the value of i was before the if structure.*

You can place more than one line between the braces, as in the code segment below.

```
if (YesNo == 'Y')
  {
    cout << "Press Enter when your printer is ready.\n";
    cin >> TempIn;
  }
```

Figure 7-5 shows the flowchart for an if structure. The if structure is sometimes called a ***one-way selection structure*** because the decision is whether to go "one way" or just bypass the code in the if structure.

Note

You may notice that a \n appears between the words of and the in the final statement of the program in Figure 7-6. This end of line character causes the word of to be the last word on the line and the to appear at the beginning of the next line on the screen. This is done because the length of the city name will vary. Forcing the sentence to break into two lines between the words of and the ensures that there is room for the city name.

PITFALLS

You have become accustomed to using semicolons to end statements. However, using semicolons to end each line in if structures can cause problems, as shown below.

```
if (i == 3);        // don't do this!!
  { cout << "The value of i is 3\n"; }
```

The statement in braces will execute in every case because the compiler interprets the semicolon as the end of the if structure.

Analyze the program in Figure 7-6. The program declares a character array of length 25 and an unsigned long integer. The user is asked for the name of their city or town, and for the population of the city or town. The if structure compares the population to a value that would indicate whether the city is among the 100 largest U.S. cities. If the city is one of the 100 largest U.S. cities, the program prints a message saying so.

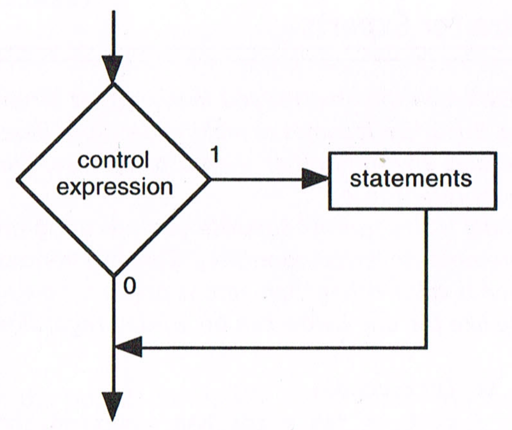

FIGURE 7-5
The if structure is sometimes called a one-way selection structure.

```
#include <iostream.h>

main()
{
 char city_name[25];
 unsigned long population;

 cout << "What is the name of your city or town? ";
 cin.get(city_name, 25);
 cin.ignore(80,'\n');

 cout << "What is the population of the city or town? ";
 cin >> population;

 if (population >= 171439)
  {
    cout << "According to the 1990 census, " << city_name
         << " is one of\nthe 100 largest U.S. cities.\n";
  }
 return 0;
}
```

FIGURE 7-6
This program uses a one-way selection structure.

EXERCISE 7-4 USING if

1. Open *CITY.CPP*. The program from Figure 7-6 appears without the if structure.
2. Add the if structure shown in Figure 7-6 to the program. Enter the code carefully.
3. Compile, link, and run the program. Enter your city or town to test the program.
4. If your city or town is not one of the 100 largest cities, enter Newport News, a city in Virginia with a population of 171,439.
5. Leave the source code file open for the next exercise.