Example 2.1. Odd or Even?

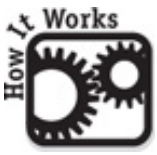
OK, enough preliminaries. It's time to look at a complete program that uses decision making. This is a simple example, but it introduces a new operator (%) and shows the **if-else** syntax in action.

The program takes a number from the keyboard and reports whether it is odd or even.

even1.cpp

```
#include <iostream>
using namespace std;

int main() {
    int  n, remainder;

    // Get a number from the keyboard.

    cout << "Enter a number and press ENTER: ";
    cin >> n;

    // Get remainder after dividing by 2.

    remainder = n % 2;

    // If remainder is 0, the number input is even.

    if (remainder == 0)
        cout << "The number is even." << endl;
    else
        cout << "The number is odd." << endl;

    system("PAUSE");
    return 0;
}
```

Once again, if you are following along and want to enter this example by hand, the comments—lines beginning with double slashes (//)—are optional.



How It Works

The first statement of the program declares two integer variables, n and remainder:

```
int  n, remainder;
```

The next thing the program does is get a number and store it in the variable n. This should look familiar by now:

```
cout << "Enter a number and press ENTER: ";
cin >> n;
```

Now it's just a matter of performing a test on n to see whether it is odd or even. How do you do that? Answer: You divide the number by 2 and look at the remainder. If the remainder is 0, the number is even (in other words, divisible by 2). Otherwise, it's odd.

That's exactly what's done here. The following statement divides by 2 and gets the remainder. This is called *modulus* or *remainder* division. The result is stored in a variable named (appropriately enough) remainder.

```
int remainder = n % 2;
```

Again, if the remainder is 0, that means n was divided evenly by 2.

The percent sign (%) loses its ordinary meaning in C++ and instead signifies "remainder division." Here are some sample results:

Example	Remainder From Division	Remarks
`3 % 2`	`1`	Odd
`4 % 2`	`0`	Even
`25 % 2`	`1`	Odd
`60 % 2`	`0`	Even
`25 % 5`	`0`	Divisible by 5
`13 % 5`	`3`	Not divisible by 5

After dividing n by 2 and getting the remainder, we get a result of either 0 (even) or 1 (odd). The **if** statement compares the remainder to 0 and prints the appropriate message.

```
if (remainder == 0)
    cout << "The number is even." << endl;
else
    cout << "The number is odd." << endl;
```

Notice the double equal signs (==) used in this code. As I mentioned earlier, test-for-equality uses double equal signs; a single equal sign (=) would mean assignment. If I'm getting repetitive on this subject, it's because when I was first learning C, I made this mistake too many times myself!

Here is the same code written in statement-compound style:

```
if (remainder == 0) {
```

```
    cout << "The number is even." << endl;
} else {
    cout << "The number is odd." << endl;
}
```

Optimizing the Code

The version of the Odd-or-Even program I just introduced is not as efficient as it could be. The remainder variable is not necessary in this case. This version is a little better:

even2.cpp

```
#include <iostream>
using namespace std;

int main() {
    int  n;

    // Get a number n from the keyboard.

    cout << "Enter a number and press ENTER: ";
    cin >> n;


    // Get remainder after dividing by 2.
    // If remainder is 0, then n is even.

    if (n % 2 == 0)
        cout << "The number is even.";
    else
        cout << "The number is odd.";

    return 0;
}
```

This version performs modulus division inside the condition, comparing the result to 0.

Exercise

Exercise 2.1.1. Write a program that reports whether a number input is divisible by 7. (Hint: If a number is divisible by 7, that means you can divide it by 7 and get a remainder of 0.)