Using braces even when they're not required has two advantages. First, the program becomes easier to modify, because more statements can easily be added to any `if` or `else` clause. Second, it helps avoid errors that can result from forgetting to use braces when adding statements to an `if` or `else` clause.

Cascaded `if` Statements

We'll often need to test a series of conditions, stopping as soon as one of them is true. A "cascaded" `if` statement is often the best way to write such a series of tests. For example, the following cascaded `if` statement tests whether `n` is less than 0, equal to 0, or greater than 0:

```
if (n < 0)
  printf("n is less than 0\n");
else
  if (n == 0)
    printf("n is equal to 0\n");
  else
    printf("n is greater than 0\n");
```

Although the second `if` statement is nested inside the first, C programmers don't usually indent it. Instead, they align each `else` with the original `if`:

```
if (n < 0)
  printf("n is less than 0\n");
else if (n == 0)
  printf("n is equal to 0\n");
else
  printf("n is greater than 0\n");
```

This arrangement gives the cascaded `if` a distinctive appearance:

```
if ( expression )
  statement
else if ( expression )
  statement
...
else if ( expression )
  statement
else
  statement
```

The last two lines (`else` *statement*) aren't always present, of course. This way of indenting the cascaded `if` statement avoids the problem of excessive indentation when the number of tests is large. Moreover, it assures the reader that the statement is nothing more than a series of tests.

Keep in mind that a cascaded `if` statement isn't some new kind of statement; it's just an ordinary `if` statement that happens to have another `if` statement as its `else` clause (and *that* `if` statement has another `if` statement as its `else` clause, *ad infinitum*).