Here's an example of a compound statement:

```
{ line_num = 0; page_num++; }
```

For clarity, I'll usually put a compound statement on several lines, with one statement per line:

```
{
  line_num = 0;
  page_num++;
}
```

Notice that each inner statement still ends with a semicolon, but the compound statement itself does not.

Here's what a compound statement would look like when used inside an `if` statement:

```
if (line_num == MAX_LINES) {
  line_num = 0;
  page_num++;
}
```

Compound statements are also common in loops and other places where the syntax of C requires a single statement, but we want more than one.

The `else` Clause

An `if` statement may have an `else` clause:

`if` statement with `else` clause

`if` (*expression*) *statement* `else` *statement*

The statement that follows the word `else` is executed if the expression in parentheses has the value 0.

Here's an example of an `if` statement with an `else` clause:

```
if (i > j)
  max = i;
else
  max = j;
```

Notice that both "inner" statements end with a semicolon.

When an `if` statement contains an `else` clause, a layout issue arises: where should the `else` be placed? Many C programmers align it with the `if` at the beginning of the statement, as in the previous example. The inner statements are usually indented, but if they're short they can be put on the same line as the `if` and `else`:

```
if (i > j) max = i;
else max = j;
```