The Structure of `if` Statements

The `if` construct allows the programmer to specify that a statement should be executed under some circumstances and not executed under others. It allows the program to make decisions: "If this condition is true, then do that." The syntax is:

```
if (boolean condition)
{
  statement(s) to be executed if condition is true
}
else
{
  statement(s) to be executed if condition is false
}
```

Some remarks: the braces are optional if there is only a single statement, and the `else` clause is always optional. The condition must be some expression that evaluates to either a zero value (false) or a non-zero value (true). In C++, nearly every expression evaluates to some value—even assignments, which evaluate to the right-hand value of the statement. The condition must always be placed in parentheses.

Example

```
if (a < b)
{
  cout << "Hey, " << a << " is less than " << b << "!\n";
}
else
{
  cout << a << " must be greater than or equal to " << b << "!\n";
}
```

Boolean Operator	Meaning of test
==	equal ?
!=	not equal ?
<=	less than or equal to ?
>=	greater than or equal to ?
<	less than ?
>	greater than ?
&&	and
||	or

if-else Ladder

Sometimes there are multiple dependent conditions that must be evaluated to determine the action to be executed. The syntax is:

```
if (boolean condition1)
{
  statement(s) to be executed if condition1 is true
}
else if (boolean condition2)
{
  statement(s) to be executed if condition2
  is true (and condition1 is false)
}
else if (boolean condition3)
{
  statement(s) to be executed   if condition3
  is true (and condition1 and condition2 are false)
}
```

Example

```
if ((time >= 8) && (day == weekday))
{
  cout << "You are late for school!" << endl;
}
else if ((day == weekend) || (time > 3))
{
  cout << "Sleep as much as you want." << endl;
}
else
{
  cout << "Enjoy your classes." << endl;
}
```