Logical Expressions and if-else statements

C++ Notes Mrs. Alano

The general form of the **if-else** statement in C++ is:

```
if (condition)
     statement1;
else
     statement2;
```

Where **condition** is a logical expression and *statement1* and *statement2* are either simple statements or **compound statements** (blocks surrounded by braces). The *else* clause is optional, so the *if* statement can be used by itself:

```
if (condition)
     statement1;
```

When an *if-else* statement is executed, the program evaluates the condition and then executes statement1 if the condition is true, and statement2 if the condition is false. When *if* is coded without the *else*, the program evaluates the condition and executes statement1 if the condition is true. If the condition is false, the program skips *statement1.*

Relational Operators:

C++ recognizes six relational operators:

Operator	Meaning
>	Is greater than
<	Is less than
>=	Is greater than or equal to
<=	Is less than or equal to
==	Is equal
!=	Is not equal

NOTE

In C++ the "is equal" condition is expressed by the "==" operator, while a single "=" means assignment. Inadvertently writing = instead of == renders your conditional statement worse than meaningless, without generating a syntax error. (Although, I think the newest version of Visual C++ does generate an error)

Logical Operators:
C++ has two binary logical operators, “and” and “or”, and one unary logical operator “not”.

Operator	Meaning
`&&`	and
`||`	or
!	not

The expression:

```
condition1 && condition2
```
Is true if and only if BOTH condition1 and condition2 are true.

```
condition1 || condition2
```
Is true if EITHER condition1 OR condition2 or both are true.

```
!condition1
```
Is true if and only if condition1 is false.

NOTE
When coding, a condition must be a complete expression that itself would evaluate to true or false

So when checking to see if a character is upper or lower case y, you can't do this

```
if (letter == 'y' || 'Y')
```

The computer reads each side of the OR separately and will evaluate this based on the ASCII value of 'Y' and not whether letter contains 'Y'

This is the correct way to use a compound expression

```
if (letter == 'y' || letter == 'Y')
```