**Logical operators.**

( !, &&, || )

The operator ! is the C++ operator for the Boolean operation NOT. It has only one operand, to its right, and inverts it, producing false if its operand is true, and true if its operand is false. Basically, it returns the opposite Boolean value of evaluating its operand. For example:

|  |  |  |
| --- | --- | --- |
| 1 2 3 4 | !(5 == 5) // evaluates to false because the expression at its right (5 == 5) is true  !(6 <= 4) // evaluates to true because (6 <= 4) would be false  !true // evaluates to false  !false // evaluates to true |  |

The logical operators && and || are used when evaluating two expressions to obtain a single relational result. The operator && corresponds to the Boolean logical operation AND, which yields true if both its operands are true, and false otherwise. The following panel shows the result of operator && evaluating the expression a&&b:

|  |  |  |
| --- | --- | --- |
| **&& OPERATOR (and)** | | |
| **a** | **b** | **a && b** |
| true | true | true |
| true | false | false |
| false | true | false |
| false | false | false |

The operator || corresponds to the Boolean logical operation OR, which yields true if either of its operands is true, thus being false only when both operands are false. Here are the possible results of a||b:

|  |  |  |
| --- | --- | --- |
| **|| OPERATOR (or)** | | |
| **a** | **b** | **a || b** |
| true | true | true |
| true | false | true |
| false | true | true |
| false | false | false |

For example:

|  |  |  |
| --- | --- | --- |
| 1 2 | ( (5 == 5) && (3 > 6) ) // evaluates to false ( true && false )  ( (5 == 5) || (3 > 6) ) // evaluates to true ( true || false ) |  |

When using the logical operators, C++ only evaluates what is necessary from left to right to come up with the combined relational result, ignoring the rest. Therefore, in the last example ((5==5)||(3>6)), C++ evaluates first whether 5==5 is true, and if so, it never checks whether 3>6 is true or not. This is known as *short-circuit evaluation*, and works like this for these operators:

|  |  |
| --- | --- |
| **operator** | **short-circuit** |
| && | if the left-hand side expression is false, the combined result is false (the right-hand side expression is never evaluated). |
| || | if the left-hand side expression is true, the combined result is true (the right-hand side expression is never evaluated). |

This is mostly important when the right-hand expression has side effects, such as altering values:

|  |  |  |
| --- | --- | --- |
|  | if ( (i<10) && (++i<n) ) { /\*...\*/ } // note that the condition increments i |  |

Here, the combined conditional expression would increase *i* by one, but only if the condition on the left of && is true, because otherwise, the condition on the right-hand side (++*i*<*n*) is never evaluated.