C++: Relational and Logical Operators *(Gaddis Chapter#4)*

RELATIONAL OPERATORS: Relational operators allow you to compare numeric and char values and determine whether one is greater than, less than, equal to, or not equal to another.

Relational Operators

C++ Symbol	Relationship
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
==	Equal to
! =	Not Equal to

Sample statements below include Relational Expressions.
(Assume x is 11, y is 8, and z is an int or bool data type)

Statement	Outcome
z = x < y	z is assigned 0 because x is not less than y.
cout << (x > y);	Displays 1 because x is greater than y.
z = (x >= y);	z is assigned 1 because x is greater than or equal to y.
cout << (x <= y);	Displays 0 because x is not less than or equal to y.
z = (y != x);	z is assigned 1 because y is not equal to x.
cout << (x == (y + 3));	Displays 1 because x is equal to y +3.

***NOTE: Relational operators are binary operators with left-to-right associativity.**

LOGICAL OPERATORS: Logical operators connect two or more relational expressions into one or reverse the logic of an expression

Logical or Boolean Operators

C++ Symbol	Meaning	Effect
&&	AND	Connects two expressions into one. Both expressions must be true for the overall expression to be true
||	OR	Connects two expressions into one. One or both expressions must be true for the overall expression to be true. It is only necessary for one to be true, and it does not matter which.
!	NOT	Reverses the "truth" of an expression. It makes a true expression false, and a false expression true.

***NOTE: The Logical AND and Logical OR operations below lists all the possible combinations of values that two expressions may have, and the resulting value returned by the use of the && or the || connecting the two expressions.**

Logical AND

Expression	Value of the expression
false && false	false (0)
false && true	false (0)
true && false	false (0)
true && true	true (1)

NOTE: If the sub-expression on the left side of an && operator is false, the expression on the right side will not be checked. Because the entire expression is false if even just one of the sub-expression is false, it would waste CPU time to check the remaining expression. This is called *short circuit evaluation*.

Logical NOT

Expression	Value of the expression
!false	true (1)
!true	false (0)

Logical OR

Expression	Value of the expression
false || false	false (0)
false || true	true (1)
true || false	true (1)
true || true	true (1)

NOTE: The || operator also performs *short circuit evaluation*. If the sub-expression on the left side of an || operator is true, the expression on the right side will not be checked. Because it is only necessary for one of the sub-expressions to be true for the whole expression to evaluate to true, it would waste CPU time to check the remaining expression. There is no || key on the computer keyboard. Use two | symbols. This symbol is on the backslash key. Press Shift and backslash to type it.

Precedence of C++ operators we have seen so far ….

Operator	
++ -- (postfix)	***Highest***
!	
* / %	
+ -	
<< >>	
< <= > >=	
= = !=	
&&	
||	
=	***Lowest***