

# Chapter 15 – Comparisons

## 1. Boolean Logic

The name given to the logic used in comparing the values of two expressions is Boolean logic, after its inventor, George Boole. In Boolean logic, the values of expressions are compared or altered using a set of operators that produce the results of *true* or *false*. In the C++ programming language, Boolean expressions return a value of type *bool*. Objects or expressions of type *bool* can only take on or return the values *true* or *false*. In the C programming language there is no type *bool*. Boolean expressions in C return the integer values 1 (for true) or 0 (for false). C++ also supports 1 as *true* and 0 as *false*.

## 2. Comparison Operators

Comparison operators in C++ require expression of like type to be placed on either side of a comparison operator. The resulting Boolean expression must be placed in parenthesis. This has the form:

*( expression comparison-operator expression )*

Basic Boolean expressions are formed using the following standard comparison operators:

Operator	Description	Example Expressions	Result
==	Equal	(24 == 4 * 6) ( 'a' == 'c' )	1 0
!=	Not Equal	(24 != 4 * 6) ( 'a' != 'c' )	0 1
<	Less Than	(24 < 4 * 6) ( 'a' < 'c' )	0 1
>	Greater Than	(24 > 4 * 6) ( 'a' > 'c' )	0 0
<=	Less Than or Equal	(24 <= 4 * 6) ( 'a' <= 'c' )	1 1
>=	Greater Than or Equal	(24 >= 4 * 6) ( 'a' >= 'c' )	1 0

All built-in types may be compared using these operators.

### **3. Comparing Null Terminated Strings**

Not being a built-in type, null terminated strings can be compared function *strcmp* defined in *string.h*. *strcmp* requires two strings as arguments and returns a negative integer if the first string is less than the second string, 0 if the strings are equal, and a positive integer if the first string is greater than the second number. They are compared from left to right based on the ASCII collating sequence. For example:

```
strcmp("Fred", "Sam")
    returns a negative integer
strcmp("Ted", "Ted")
    returns 0
strcmp("Ted", "Sam")
    returns a positive integer
```

### **4. Comparing String Class Objects**

*String class* objects can be compared with the C++ comparison operators. Like null terminated strings, they are compared based on the ASCII collating sequence.

### **5. Logical Operators**

Boolean expressions can be combined by the use of the following logical operators:

Operator	Description
&&	And
	Or

Using these operators, two or more Boolean expressions can be combined to form one expression that resolves to 0 or 1. When the && operator is used, the Boolean expressions on both sides of && must be true for the whole expression to be true. When the || operator is used, one or both expressions being true will result in the whole expression being true. This is expressed in the following tables:

True && True	True (1)
True && False	False (0)
False && True	False (0)
False && False	False (0)

True    True	True (1)
True    False	True (1)
False    True	True (1)
False    False	False (0)

Here are several examples:

$(35 / 7 == 5 \ \&\& \ 'a' < 'c')$   
*returns 1*

$(22 \% 3 < 2 \ || \ 45 > 99)$   
*returns 1*

$(22 \% 3 < 2 \ \&\& \ 45 > 99)$   
*returns 0*

### **Exercises**

- 1) Evaluate each of the following Boolean expressions to determine if they are true or false.
  - a)  $(88 \geq 77)$
  - b)  $('a' > 'A')$
  - c)  $(45 * 3 \leq 137)$
  - d)  $!(45 * 3 \leq 137)$
  - e)  $(45 * 3 \leq 137 \ \&\& \ -3 < -6)$
  - f)  $(9 > 5 \ \&\& \ (45 * 3 \leq 137 \ || \ -3 < -6))$
- 2) To what do the following evaluate?
  - a) `strcmp("Bill", "Ted");`
  - b) `strcmp("Bill", "Abe");`
  - c) `strcmp("Bill", "Bill");`
- 3) Find and correct the errors in each of the following.
  - a) `strcmp("Bill, "Ted");`
  - b)  $(88 \Rightarrow 77)$
  - c)  $(a > 'A')$
  - d)  $(45 * 3 \Rightarrow 137)$
  - e)  $!(45 * 3 \leq 137)$
  - f)  $(45 * 3 \leq 137 \ \& \ -3 < -6)$
  - g)  $(9 > 5 \ \&\& \ (45 * 3 \leq 137 \ \text{or} \ -3 < -6))$