### C++ Header files & Operators

In this C++ tutorial, we will talk about header files and operators. In our last lesson, we discussed the basic input and output. Lets now cover header files and operators in C++ language:

#### Header Files in C++:

"**#include"** is used in C++ to import header files in our C++ program. The reason to introduce the**"**<**iostream**>" header file into our program is that functions like "**cout"** and "**cin"** are defined in that header file. There are two types of header files:

##### **1.System Header Files:**

System header files ships with the compiler. For example, “**#include <iostream>**”. To see the references for header files click [here](https://en.cppreference.com/w/cpp/header)

##### **2.User-Defined Header Files:**

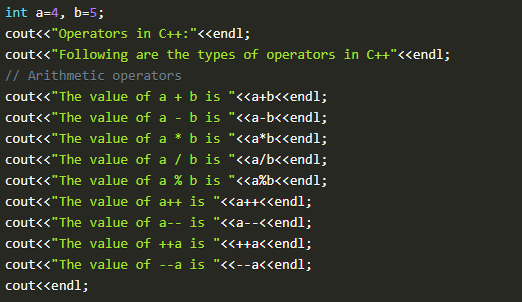
The programmer writes User-defined header files himself. To include your header file in the program, you first need to make a header file in the current directory, and then you can add it.

#### Operators in C++

Operators are used for producing output by performing various types of calculations on two or more inputs. In this lecture, we will see the operators in C++.

##### **1.Arithmetic Operators:**

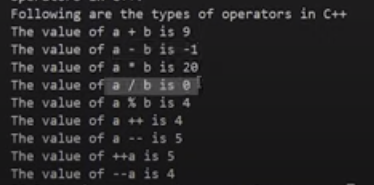
Arithmetic operators are used for performing mathematical operations like (+, -, \*). The arithmetic operators are shown in Figure 1.



***Figure 1: Arithmetic Operators***

1. The function "**a+b"**, will add a and b values and print them.
2. The function "**a-b "**will subtract a and b values and print them.
3. The function "**a\*b"** will multiply a and b values and print them.
4. The function "**a/b "**, will divide a and b values and print them.
5. The function "**a%b "**, will take the modulus of a and b values and print them.
6. The function "**a++"** will first print the value of a and then increment it by 1.
7. The function "**a--"**, will first print the value of a and then decrement it by 1.
8. The function "**++a"**, will first increment it by one and then print its value.
9. The function "**--a"**, will first decrement it by one and then print its value.

The output of these arithmetic operators is shown in figure 2.



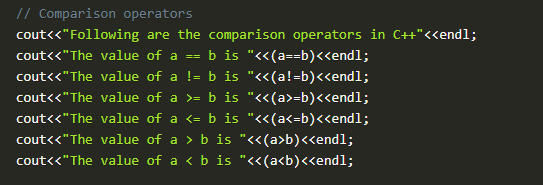
***Figure 2: Arithmetic Operators Output***

##### **2.Assignment Operators:**

Assignment operators are used for assigning values to variables. For example: **int a = 10, b = 5;**

##### **3.Comparison Operators:**

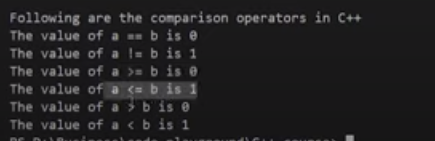
Comparison operators are used for comparing two values. Examples of comparison operators are shown in figure 3.



***Figure 3: Comparison Operators***

1. The function "**(a==b)"**, will compare a and b values and check if they are equal. The output will be one if equal, and 0 if not.
2. The function "**(a!=b)"**, will compare a and b values and check if "a" is not equal to "b". The output will be one if not equal and 0 if equal.
3. The function "**(a>=b)"**, will compare a and b values and check if "a" is greater than or equal to "b". The output will be one if greater or equal, and 0 if not.
4. The function "**(a<=b)"**, will compare a and b values and check if "b" is greater than or equal to "a". The output will be one if greater or equal, and 0 if not.
5. The function "**(a>b)"**, will compare a and b values and check if "a" is greater than "b". The output will be one if greater and 0 if not.
6. The function "**(a<b)"**, will compare a and b values and check if "b" is greater than "a". The output will be one if greater and 0 if not.

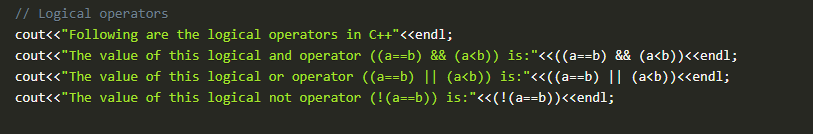
The output of these comparison operators is shown in figure 4.



***Figure 4: Comparison Operators Output***

##### **4.Logical Operators:**

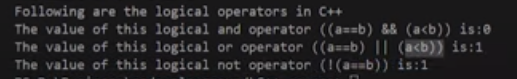
Logical operators are used for comparing two expressions. For example ((a==b) && (a>b)). More examples of logical operators are shown in figure 5.



***Figure 5: Logical Operators***

1. The function "(**(a==b)&& (a<b))"** will first compare a and b values and check if they are equal or not; if they are equal, the next expression will check whether "a" is smaller than "b". The output will be one if both expressions are true and 0 if not.
2. The function "(**(a==b) || (a<b))"**, will first compare a and b values and check if they are equal or not, even if they are not equal it will still check the next expression ie whether "a" is smaller than "b" or not. The output will be one if any one of the expressions is true and 0 if both are false.
3. The function "(!**(a==b))"**, will first compare a and b values and check if they are equal or not. The output will be inversed ie if "a" and "b" are equal; the output will be 0 and 1 otherwise.

The output of these logical operators is shown in figure 6.



***Figure 6: Logical Operators Output***

That's it for this tutorial. In this lecture, we have covered some important operators in C++ language, but there are still some operators left, which we will cover in upcoming tutorials.

#### Code as described/written in the video

// There are two types of header files:

// 1. System header files: It comes with the compiler

#include<iostream>

// 2. User defined header files: It is written by the programmer

// #include "this.h" //--> This will produce an error if this.h is no present in the current directory

using namespace std;

int main(){

int a=4, b=5;

cout<<"Operators in C++:"<<endl;

cout<<"Following are the types of operators in C++"<<endl;

// Arithmetic operators

cout<<"The value of a + b is "<<a+b<<endl;

cout<<"The value of a - b is "<<a-b<<endl;

cout<<"The value of a \* b is "<<a\*b<<endl;

cout<<"The value of a / b is "<<a/b<<endl;

cout<<"The value of a % b is "<<a%b<<endl;

cout<<"The value of a++ is "<<a++<<endl;

cout<<"The value of a-- is "<<a--<<endl;

cout<<"The value of ++a is "<<++a<<endl;

cout<<"The value of --a is "<<--a<<endl;

cout<<endl;

// Assignment Operators --> used to assign values to variables

// int a =3, b=9;

// char d='d';

// Comparison operators

cout<<"Following are the comparison operators in C++"<<endl;

cout<<"The value of a == b is "<<(a==b)<<endl;

cout<<"The value of a != b is "<<(a!=b)<<endl;

cout<<"The value of a >= b is "<<(a>=b)<<endl;

cout<<"The value of a <= b is "<<(a<=b)<<endl;

cout<<"The value of a > b is "<<(a>b)<<endl;

cout<<"The value of a < b is "<<(a<b)<<endl;

// Logical operators

cout<<"Following are the logical operators in C++"<<endl;

cout<<"The value of this logical and operator ((a==b) && (a<b)) is:"<<((a==b) && (a<b))<<endl;

cout<<"The value of this logical or operator ((a==b) || (a<b)) is:"<<((a==b) || (a<b))<<endl;

cout<<"The value of this logical not operator (!(a==b)) is:"<<(!(a==b))<<endl;

return 0;

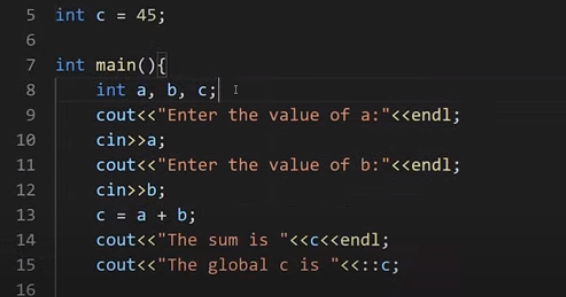
}

### C++ Reference Variables & Typecasting :

* **Built-in Data Types**
* **Float, Double and Long Double Literals**
* **Reference Variables**
* **Typecasting**

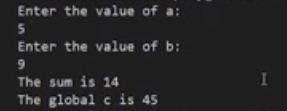
#### 1.Built-in Data Types:

As discussed in our previous lectures, built-in data types are pre-defined by the language and can be used directly. An example program for built-in data types is shown in figure 1.



***Figure 1: Built-in Data Types***

The code of built-in data types can be seen in figure 1 where we have declared three variables "**a, b and c"** inside the main function and one variable "**c"** outside the main function which is a global variable. To access the value of the global variable "**c**," we use **scope resolution operator** "**::"** with the "**c"** variable. The output of the following program is shown in figure 2.

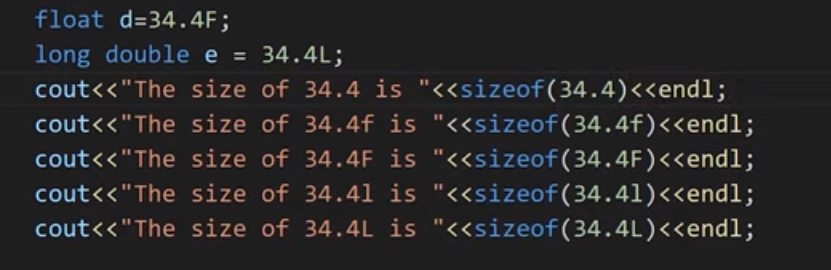


***Figure 2: Built-in Data Types Output***

As we have entered the value of the variable "**a"** as five and "**b"** as 6, it gives us the sum 14, but for the global variable, it has given us the value **45.**

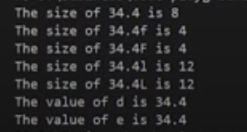
#### Float, Double and Long Double Literals:

The main reason to discuss these literals was to tell you an important concept about them. The float, double and long double literals program is shown in figure 3.



***Figure 3: Float, Double & Long Double Literals***

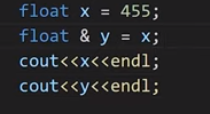
So the crucial concept which I am talking about is that in C++ language, double is the default type given to a  decimal literal (34.4 is double by default and not float), so to use it as float, you have to specify it like "**34.4F**," as shown in figure 3. To display the size of float, double, and long double literals, we have used a "**sizeof"** operator. The output of this program is shown in figure 4.



***Figure 4: Float, Double, Long Double Literal Output***

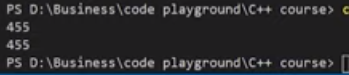
#### Reference Variable:

Reference variables can be defined as another name for an already existing variable. These are also called an alias. For example, let us say we have a variable with the name of "**sum",** but we also want to use the same variable with the name of "**add"**, to do that we will make a reference variable with the name of "**add"**. The example code for the reference variable is shown in figure 5.



***Figure 5: Reference Variable Code***

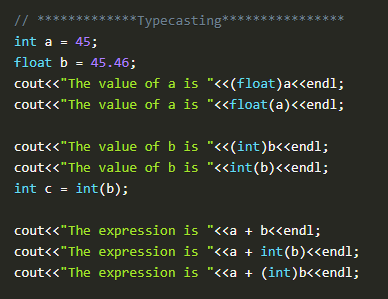
As shown in figure 5, we initialized a variable "**x"** with the value "**455"**. Then we assigned the value of "**x"** to a reference variable "**y"**. The ampersand "**&"** symbol is used with the "**y"** variable to make it reference variable. Now the variable "**y"** will start referring to the value of the variable "**x"**. The output for variable "**x"** and "**y"** is shown in figure 6.



***Figure 6: Reference Variable Code Output***

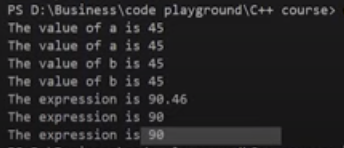
#### Typecasting:

Typecasting can be defined as converting one data type into another. Example code for type casting is shown in figure 7.



***Figure 7: Typecasting Example Code***

As shown in figure 7, we have initialized two variables, integer "**a"** and float "**b"**. After that, we converted an integer variable "**a"** into a float variable and float variable "**b"** into an integer variable. In C++, there are two ways to typecast a variable, either using "**(float)a"** or using "**float(a)"**. The output for the above program is shown in figure 8.



***Figure 8: Typecasting Program Output***

#### Code as described/written in the video

#include<iostream>

using namespace std;

int c = 45;

int main(){

// \*\*\*\*\*\*\*\*\*\*\*\*\*Build in Data types\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// int a, b, c;

// cout<<"Enter the value of a:"<<endl;

// cin>>a;

// cout<<"Enter the value of b:"<<endl;

// cin>>b;

// c = a + b;

// cout<<"The sum is "<<c<<endl;

// cout<<"The global c is "<<::c;

// \*\*\*\*\*\*\*\*\*\*\*\*\* Float, double and long double Literals\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// float d=34.4F;

// long double e = 34.4L;

// cout<<"The size of 34.4 is "<<sizeof(34.4)<<endl;

// cout<<"The size of 34.4f is "<<sizeof(34.4f)<<endl;

// cout<<"The size of 34.4F is "<<sizeof(34.4F)<<endl;

// cout<<"The size of 34.4l is "<<sizeof(34.4l)<<endl;

// cout<<"The size of 34.4L is "<<sizeof(34.4L)<<endl;

// cout<<"The value of d is "<<d<<endl<<"The value of e is "<<e;

// \*\*\*\*\*\*\*\*\*\*\*\*\*Reference Variables\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Rohan Das----> Monty -----> Rohu ------> Dangerous Coder

// float x = 455;

// float & y = x;

// cout<<x<<endl;

// cout<<y<<endl;

// \*\*\*\*\*\*\*\*\*\*\*\*\*Typecasting\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

int a = 45;

float b = 45.46;

cout<<"The value of a is "<<(float)a<<endl;

cout<<"The value of a is "<<float(a)<<endl;

cout<<"The value of b is "<<(int)b<<endl;

cout<<"The value of b is "<<int(b)<<endl;

int c = int(b);

cout<<"The expression is "<<a + b<<endl;

cout<<"The expression is "<<a + int(b)<<endl;

cout<<"The expression is "<<a + (int)b<<endl;

return 0;

}