operator and their types in C++

Operator in C++ are following,

C++ Operator

just like a symbol, just as we use the operator for mathematical work in the real world, we use the operator to perform the same task in a program. For Example,

```
A + B
```

where **"+"** is an operator and **A, B** are operand in the Programming word we can say variables.

In the programming world, these operators are divided into different categories, also in C++. C ++ offers a different type of operators. Some of which are described below.

Here we will discuss some important Operators with Example which are given,

A + B

Arithmetic Operator in C++

These are basic operators or default operator. These operators use two variable to perform a task. normally these are present in all programming languages.

These are described in the table below

Operator	Name	Example
=	addition	a + b
–	subtraction	a – b
*	multiplication	a * b
/	division	a / b
%	modulus	a % b

In the Program will be as follows,

जैसे –

```
int a= 3,b=4;

cout<<a+b; // print addition 7

cout<<a-b; // print addition -1

cout<<a*b; // print addition 12
```

Relational Operator in C++

In C++, adding a single operator or adding two different types of operator, makes a third types operator is called Relational Operator. These operators in C++ are used to compare two variables in a single statement.

Operator	Name	Example
<	Less than	a < b
<=	Less than equal to	a <=b
>	Greater than	a > b
>=	Greater than and equal to	a >= b
==	Equal to	a == b
!=	Not equal to	a != b

relation operators will be used with the control statement. such as,

In the following a=3,b=4,

with "Less Than" Operator

less than operator

```
if(a<b) // if 3 is greater than 4 than body of if will be execute.

 {

   // body of if

 }
```

“Not Equal To” Operator

```
if(a!= b) if a and b value is not equal than body of if will be execute.

 {

  // body of if

 }
```

“Equal To” operator

```
if(a== b) if a and b value is equal than body of if will be execute.

 {

     // body of if

   }
```

यहाँ इसका program दिया गया है –

```
#include<iostream>

using namespace std;



int main()

  {

    int a = 3, b = 4 , c =4;



     if(a>b) {
```

```
         cout<<b<<" is greater than "<<a;

       }



     if(a!=b) {

         cout<<a<<" is Not Equal to "<<b;

      }



     if(b==c) {

         cout<<b<<" is Equal to "<<c;

      }



    return 0;

}
```

OUTPUT

```
4 is greater than 3

4 is Not Equal to 3

4 is Equal to 4
```

Logical Operator in C++

This type of operator used to compare more than one condition together. where the condition will be a Relational type Operator.

Operator	Name	Example
&&	AND	a > b && a != b
||	OR	a >b || a != b
!	NOT	! a

if variable **a** value is greater than variable **b** AND not equal to **b** than **body of if** will be executed, means both conditions need to be true for executing **body of loop** such as,

```
int a = 5, b = 10 c = 20;




if(a < b && a > 0)  {

  //body of if

 }
```

if variable **a** value is greater than variable **b** but not equal to **b** than **body of if** will be executed, doesn't matter if one condition is becomes false, means for executing body of loop only one condition need to be True.

```
if(a < b || a != b) {

  //body of if

 }
```

NOT operator can be used with a single variable,

```
int c = 0;



 if(!c) // if c value 0, than execute body of if

  {

   //body of if

  }
```

in the above codes " **! "** operator behaves like an **"ON"** and **"OFF"** button where **ON** is **0** and other different value will be **OFF**. Therefore, it is used mostly to execute a statement or function in the program.

```
#include<iostream>

using namespace std;



 int main()

  {

    int a = 5, b = 10 ;



   if(a < b && a > 0) {

         cout<<a<<" is smaller than "<<b<<" but greater than 0;

       }
```

```
   if(a < b || a != b) {

         cout<<a<<" and "<<b<<"are not Equal";

      }



return 0;

}
```

OUTPUT

```
5 is smaller than 10 but greater than 0

5 and 10 are Not Equal
```

Assignment operator in C++

assignment operator used to assign a value to a variables/operands in the Program, such as,

```
int a = 5;

float b = 0.5;
```

Here are some assignment operator in C++

Operator	Name	Description
=	Assignment	a = 5 assign values to operands

+=	Addition and assign	a += b // a = a + b add Left Side variable to the Right Side variable then assign the final result to the Left side variable.
-=	subtraction and ssign	a -= b // a = a – b same as above with different operation
/=	division and assign	a /= b // a = a / b same as above with different operation
%=	modulus and assign	a %= b // a = a % b same as above with different operation

In the below Program, we used **=, +=** and ***=** operator

```
#include<iostream>

using namespace std;



int main()

  {

    int count,sum=0,mul=1; //variable initialization using assignment operator
```

```
      for(count = 1; count<=5;count++)

      { sum += count; // store result left side mul *= count; // } cout<<"Adition : "<<sum; //
print variable sum value cout<<"\nMultiply: "<<mul<<endl; // print variable mul value return
0; }
```

OTUPUT

```
Addition: 15

Multiply: 120
```

just like other operators **[-=, %=, /=]** will be used.

Special Operator in C++

This is not a different category operator, but it is a combination of two different or similar operators which are used in different operations.

Here are some special Operator in C++,

Operator	Name	Description
#	Pound sign	indicates pre-processor directive.
?:	Conditional or ternary Operator	used in if-else statement.
++	increment	increase value one by one
—	decrement	decrease value one by one

&	reference operator	using reference variable
*	de-reference operator	To store the address of a variables
sizeof()	sizeof operator	To find the size of a data-type or a variable
,	comma operator	For multiple variable, initialization
::	scope resolution operator	To change the scope of a variable and for the outline-definition of their function members in class and structure.
.	dot operator	To access the members of a structure, union and class
->	arrow operator	To access the members of a structure and class using pointer
new	new operator	for dynamic memory allocation
delete	delete operator	for de-locate the memory
endl	new line feed operator	To insert a new line in the program. just like the keyboard has an enter button.

Here, some of these special operators used in the below program,

```
#include<iostream>
```

```
using namespace std;



int main()

  {

    int a =4; // assignment operator



    cout<<"Address of count variable : "<<&a<<"\n"; // reference operator

    cout<<"size of int data type(in byte): "<<sizeof(a); // sizeof operator

  return 0;

}
```

OUTPUT

```
Address of count variable : 0x8f87fff4

Size of int data type(int byte): 2
```

Example of scope resolution operator in C++

This operator is used in the global variable when a program has the same name, as the local variable and the global variable, in this case, we use the scope resolution operator **(::)** to access the value of the global variable.

You can understand this from the program given below, where two same name **num** variables have been declared in global and local and the scope operator has been used to access the value of the global declared variable-

```
#include<iostream>
```

```
using namespace std;

  int num=3; // global varaible

  int main()

   {

     int num = 6; //local variable

      cout<<"local num : "<<num<<endl; // for local variable

     cout<<"global ::num: "<<::num; // global variable

    return 0;

   }
```

OUTPUT

```
local num : 6

global ::num: 3
```

it is also used in to define member function definition outside in the class and structure. which is further explained in this tutorial.

Note: C++ has also more operators.