# 14. Relational Operators

- Shivam Malhotra

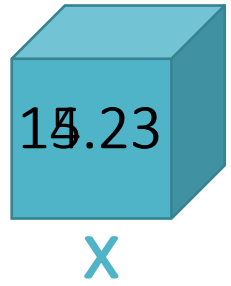
# Find Output

```
#include <iostream.h>
int main()
{
        double x = 15.23;
        cout << x-- << endl;
        cout << (x - 1) << endl;
        cout << --x << endl;
        return 0;
}
```

Let us solve it step by step

# Find Output

```
#include <iostream.h>
int main()
{
        double x = 15.23;
        cout << x-- << endl;
        cout << (x - 1) << endl;
        cout << --x << endl;
        return 0;
}
```




For postfix decrement x--,

Operation : Decrease value of x by 1

Value : Equal to original x

Hence after this statement executes,

Value of x = 14.23

And we get 15.23 in output

# Find Output

```
#include <iostream.h>
int main()
{
        double x = 15.23;
        cout << x-- << endl;
        cout << (x - 1) << endl;
        cout << --x << endl;
        return 0;
}
```


For expression (x-1),

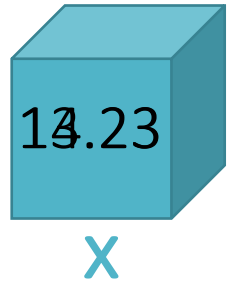
Operation : Subtract 1 from x

Value : Equal to x -1

Hence, we get 13.23 in the output

# Find Output

```
#include <iostream.h>
int main()
{
        double x = 15.23;
        cout << x-- << endl;
        cout << (x - 1) << endl;
        cout << --x << endl;
        return 0;
}
```




For prefix decrement --x,

Operation : Decrease value of x by 1

Value : Equal to (original x) - 1

Hence after this statement executes,

Value of x = 13.23

And we get 13.23 in the output

# Find Output

```
#include <iostream.h>
int main()
{
        double x = 15.23;
        cout << x-- << endl;
        cout << (x - 1) << endl;
        cout << --x << endl;
        return 0;
}
```

Final Output :

15.23

13.23

13.23

# Relational Operators

C++ provides relational operators for comparing numbers and characters

There are 6 relational operators

- Less than : a < b
- Less than or equal to : a <= b
- Greater than : a > b
- Greater than or equal : a >= b
- Equal to : a == b
- Not equal to operator : a != b

The value of relational expression is 0 if false and 1 if true.

Ex. Expression (15<20) will compare 15 with 20 and the value of expression will be 1 since it is true.

# < and <=

- For expression (a **<** b)

**Operation** : Check if a is less than b

**Value** : 1 if true, 0 if false

- For expression (a **<=** b)

**Operation** : Check if a is less than or equal to b

**Value** : 1 if true, 0 if false

```
int x = 15;
int y = 15;
x = ( x < y ) ;
y = ( x <= y ) ;
cout << x << endl << y;
```

**Output:**

0

1

# > and >=

- For expression (a **>** b)

**Operation** : Check if a is greater than b

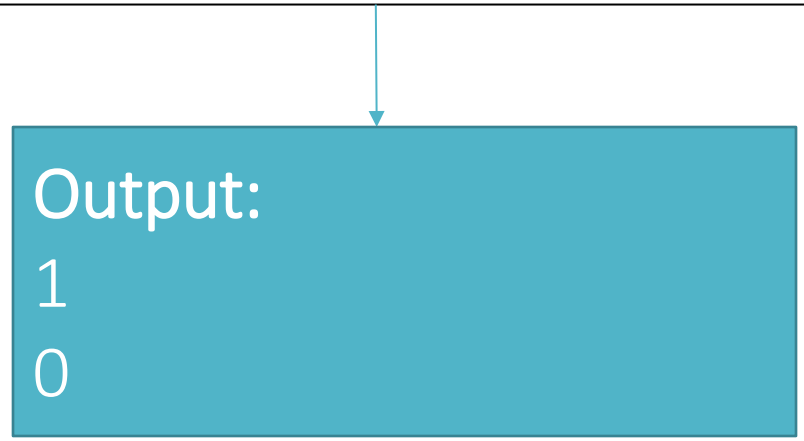
**Value** : 1 if true, 0 if false

- For expression (a **>=** b)

**Operation** : Check if a is greater than or equal to b

**Value** : 1 if true, 0 if false

```
int x = 15;
int y = 15;
x = ( x >= y ) ;
y = ( x > y ) ;
cout << x << endl << y;
```



# == and !=

- For expression (a **==** b)

**Operation** : Check if a is equal to b

**Value** : 1 if true, 0 if false

- For expression (a **!=** b)

**Operation** : Check if a is not equal to b

**Value** : 1 if true, 0 if false

```
int x = 15;
int y = 15;
x = ( x == y ) ;
y = ( x != y ) ;
cout << x << endl << y;
```

Output:
1
1

# Assignment operator (=)

For expression (a = b)

**Operation** : Assign value of b to a

**Value** : Equal to value of b

Consider the statement : `x = ( y = 15 ) ;`

Compare it with expression : `x = ( y + 15 ) ;`

Thus in our original expression, first 15 will be assigned to y, and then the value of expression ( y = 15 ) will be assigned to x ( which is 15 only)

# Common Mistake

Do not confuse == operator with = operator

For example,

- (x = 15) will assign 15 to x and the value of expression will be 15
- (x == 15) will compare if x is equal to 15 and the value of expression will be 0 or 1 depending on the comparison result

# What’s ahead?

In the next video, we will study about logical operators