ARITHMETIC OPERATORS

The five arithmetical operations supported by C++ language are:

+	addition	Correspond with their respective mathematical operators
-	subtraction	
*	multiplication	
/	division	
%	modulo	Is the operation that gives the remainder of a division of two values.

```
#include <iostream>
using namespace std;

main()
{
   int a = 21;
   int b = 10;
   int c ;

   c = a + b;
   cout << "Line 1 - Value of c is :" << c << endl ;
   c = a - b;
   cout << "Line 2 - Value of c is  :" << c << endl ;
   c = a * b;
   cout << "Line 3 - Value of c is :" << c << endl ;
   c = a / b;
   cout << "Line 4 - Value of c is  :" << c << endl ;
   c = a % b;
   cout << "Line 5 - Value of c is  :" << c << endl ;
   return 0;
}
```

Increase and decrease
The increase operator and the decrease operator increase or reduce by one the value stored in a variable.
(++) the increase operator a++ equivalent to a=a+1
(--) the decrease operator a— equivalent to a=a-1

```
#include <iostream>
using namespace std;

main()
{
   int a = 21;
   int b = 10;
   int c ;

   c = a++;
   cout << "Line 6 - Value of c is :" << c << endl ;
   c = a--;
   cout << "Line 7 - Value of c is  :" << c << endl ;
   return 0;
}
```

Relational and equality operators

In order to evaluate a comparison between two expressions we can use the relational and equality operators. The result of a relational operation can only be TRUE or FALSE.

= =	Equal to
!=	Not equal to
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to

Instead of using only numeric constants, we can use any valid expression, icluding variables.

Be careful! The operator = is not the same as the operator = =. The fisrt one is an assignment operator (assigns the value at its right to the variable at its left) and the other one is the equality operator that compares whether both expressions in the two sides of it are equal to each other.

```
#include <iostream>
using namespace std;

main()
{
   int a = 21;
   int b = 10;
   int c ;

   if( a == b )
   {
      cout << "Line 1 - a is equal to b" << endl ;
   }
   else
   {
      cout << "Line 1 - a is not equal to b" << endl ;
   }
   if ( a < b )
   {
      cout << "Line 2 - a is less than b" << endl ;
   }
   else
   {
      cout << "Line 2 - a is not less than b" << endl ;
   }
   if ( a > b )
   {
      cout << "Line 3 - a is greater than b" << endl ;
   }
   else
   {
      cout << "Line 3 - a is not greater than b" << endl ;
   }
   /* Let's change the values of a and b */
   a = 5;
   b = 20;
   if ( a <= b )
   {
      cout << "Line 4 - a is either less than \
                                       or euqal to  b" << endl ;
   }
   if ( b >= a )
   {
      cout << "Line 5 - b is either greater than \
                                       or equal to b" << endl ;
   }
   return 0;
}
```

Logical operators

! - is performing operation NOT, is located on the right , and the only thing that it does is to inverse the value from TRUE to FALSE or from FALSE to TRUE

&& and **ll** **-** are used when evaluating two expressions to obtain a single relational result.
&& correspond of AND
ll corresponds of OR

```
#include <iostream>
using namespace std;

main()
{
   int a = 5;
   int b = 20;
   int c ;

   if ( a && b )
   {
      cout << "Line 1 - Condition is true"<< endl ;
   }
   if ( a || b )
   {
      cout << "Line 2 - Condition is true"<< endl ;
   }
   /* Let's change the values of  a and b */
   a = 0;
   b = 10;
   if ( a && b )
   {
      cout << "Line 3 - Condition is true"<< endl ;
   }
   else
   {
      cout << "Line 4 - Condition is not true"<< endl ;
   }
   if ( !(a && b) )
   {
      cout << "Line 5 - Condition is true"<< endl ;
   }
   return 0;
}
```

Other operators

Later on the next level programme ????? (numele dat programului), we will see a few more operations.

Precedence of operators

If you want to write complicated expressions and you are not completely sure of the precedent levels, always include parentheses. It will also easier to read.

```
int a=10+(5*(32*45-64)-2)*3;
```