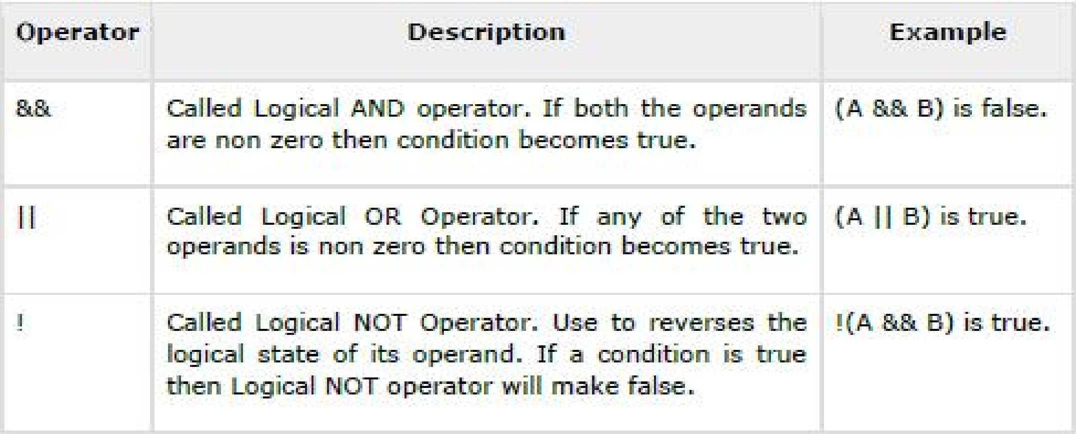
C# Logical Operator

The C# Logical Operator also evaluates the values and returns true or false as output. Based on true-false the program behaves dynamically at runtime.

Types of Logical Operators:

- AND (&&)
- OR (||)
- NOT (!)



Operator	Description	Example
&&	Called Logical AND operator. If both the operands are non zero then condition becomes true.	(A && B) is false.
||	Called Logical OR Operator. If any of the two operands is non zero then condition becomes true.	(A || B) is true.
!	Called Logical NOT Operator. Use to reverses the logical state of its operand. If a condition is true then Logical NOT operator will make false.	!(A && B) is true.

&& Operator

- It is pronounced as **AND operator**.
- It returns true if both or all the conditions are true and return false if any of the condition is false.

Sample Program

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace and_operator
{
    class Program
    {
        static void Main(string[] args)
        {
            string name, password;

            name = "Steven";
            password = "Steven123";

            // evaluating both expresson and returns true if         all are true.
            if (name == "Steven" && password == "Steven123")
            {
                Console.WriteLine("Login Successful");
            }
            else
            {
                Console.WriteLine("Unauthorised access");
            }
            Console.ReadLine();
        }
    }
}
```

Output

```
Login Successful_
```

|| Operator

- It is pronounced as **or operator**.
- It also returns true or false based on condition. If any one of the condition matches, then it returns true but if both or all the conditions are false then it returns false.

Sample Program

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Or_operator
{
    class Program
    {
        static void Main(string[] args)
        {
            string username, userpassword;

        label: //Creating label

            Console.Write("\n\nEnter your login name:\t");
            username = Console.ReadLine();

            Console.Write("Enter your password:\t");
            userpassword = Console.ReadLine();

            try
            {
                if ((username == "Steven" || username == "Clark") && (userpassword == "demopass"))
                {
                    Console.WriteLine("\nLogin Successful.");
                }
                else
                {
                    Console.WriteLine("\nUnauthorised Access. Aborting...");
                }

                Console.Write("\n\nPress Y or y for continue. Press N or n for Exit:\t");
                char ans = Convert.ToChar(Console.ReadLine());
                if (ans == 'Y' || ans == 'y')
                {
                    goto label; //goto label

                }
                else
                {
                    Console.WriteLine("Press  Enter for Aborting...");
                    Console.ReadLine();
                    return;
                }
            }
            catch { }

        }
    }
}
```

Output

```
Enter your login name :  Steven
Enter your password :    demopassLogin
Successful.Press Y or y for continue. Press N or n for Exit:    yEnter your login name :
   Clark
Enter your password :     demopass

Login Successful.

Press Y or y for continue. Press N or n for Exit:    y

Enter your login name :   Mark
Enter your password :     demopass



Unauthorised Access. Aborting...
```

```
Press Y or y for continue. Press N or n for Exit:    n
Press Enter for Aborting...
_
```

! Operator

- It is pronounced as **not operator**.
- Use to reverse the logical state of its operand. It returns true if expression is false and returns false if expression is true

Examples:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Not_Operator
{
    class Program
    {
        static void Main(string[] args)
        {
            string username, password;

            Console.Write("Enter user name:\t");
            username = Console.ReadLine();
            Console.Write("Enter Password:\t");
            password = Console.ReadLine();

            if (!(username == "Steven" && password == "demopass"))
            {
                Console.WriteLine("\nLogin Successful");
            }
            else
            {
                Console.WriteLine("\nUnauthorised Access. Aborting...");
            }
            Console.ReadLine();
        }
    }
}
```

In the above example if you will enter the username as Steven and Password as demopass then it will deny you to login because it is evaluated by **not (!) operator** that returns false if the conditions match.

Output

```
Enter user name :   Steven
Enter Password :  demopass
Unauthorised Access. Aborting... _
```

^ Operator

- It is pronounced as **xor operator**.
- It returns false if the following condition matches:

  (i) if both or all the expression returns true.

  (ii) If both or all the expression returns false.

Sample Program

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace xor_operator
{
    class Program
    {
        static void Main(string[] args)
        {
            string name, password;

            name = "Steven";
            password = "demopass";

            //it returns false because both expression match.
            if ((name == "Steven") ^ (password == "demopass"))
            {
                Console.WriteLine("Acess granted...");
            }
            else
            {
                Console.WriteLine("Access Denied. Aborting...");
            }
            Console.ReadLine();
        }
    }
}
```

In the above example, the both expression returns true, so the xor operator returns false and print "Access Denied. Aborting…"

Output

```
Access Denied. Aborting..._
```