

C# Switch Case Statement with Examples

In **C#**, **Switch** is a selection statement, and it will execute a single case statement from the list of multiple case statements based on the pattern match with the defined expression.

Using the switch statement in **C#**, we can replace the functionality of `if...else if` (</tutorial/csharp/csharp-if-else-if-statement-with-examples>) statement to provide better readability for the code.

Syntax of C# Switch Statement

Generally, in **C#** switch statement is a collection of multiple case statements, and it will execute only one single case statement based on the matching value of an expression.

Following is the syntax of defining the switch statement in the **C#** programming language.

```
switch(variable/expression){  
    case value1:  
        // Statements to Execute  
        break;  
    case value2:  
        //Statements to Execute  
        break;  
    ....  
    ....  
    default:  
        // Statements to Execute if No Case Matches  
        break;  
}
```

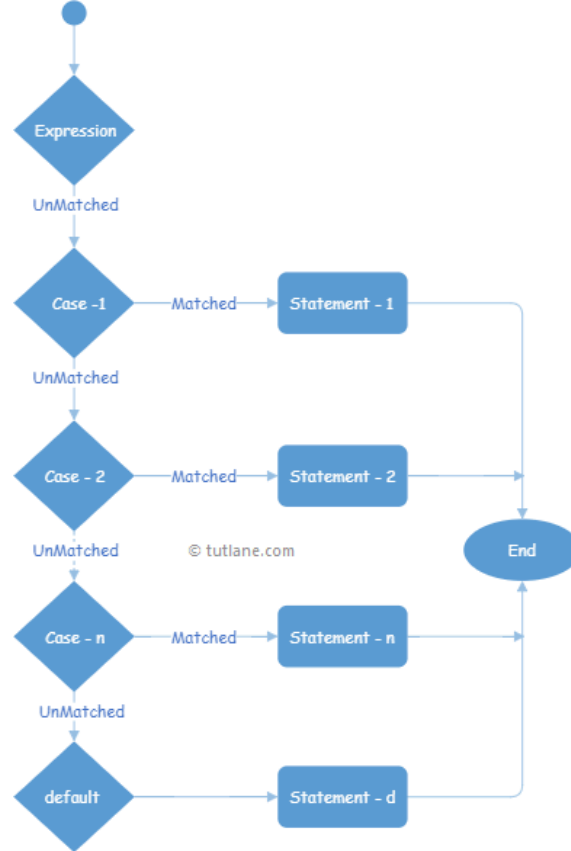
If you observe the above syntax, we defined a **switch** statement with multiple **case** statements. Here, the **switch** statement will evaluate the **expression / variable** value by matching the **case** statement values (value1, value2, etc.). If the **variable/expression** value matches with any **case** statement, the statements inside of the particular **case** will be executed.

If none of the **case** statements are matched with the defined **expression/variable** value, then the statements inside of the **default** block will be executed, and it's more like an **else** block in the `if...else` (</tutorial/csharp/csharp-if-else-statement-with-examples>) statement.

If you observe the above code, we used a **break** keyword at the end of each **case** statement to stop the further execution of non-matching **case** statements in the switch.

C# Switch Statement Flow Chart Diagram

Following is the pictorial representation of the switch case statement process flow in the **C#** programming language.



If you observe the above switch statement flow chart, the switch statement's process flow will start from **Top** to **Bottom**, and in the first case, it will check whether the expression value matches or not.

In case the expression value matches mean it will execute the particular **case** statements block and exist the **switch** statement; otherwise, it will go to the second **case** statement and check whether the expression value matching or not, the same way search will continue till it finds the right **case** statement.

If all **case** statements fail to match the defined expression value, then the **default** block statements will be executed, and the switch statement will come to an end.

C# Switch Case Statement Example

Following is the example of using switch statements in the c# programming language.

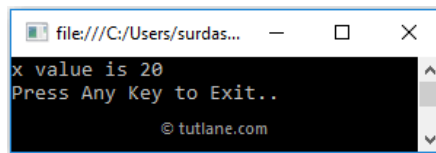
```

using System;

namespace Tutlane
{
    class Program
    {
        static void Main(string[] args)
        {
            int x = 20;
            switch (x)
            {
                case 10:
                    Console.WriteLine("x value is 10");
                    break;
                case 15:
                    Console.WriteLine("x value is 15");
                    break;
                case 20:
                    Console.WriteLine("x value is 20");
                    break;
                default:
                    Console.WriteLine("Not Known");
                    break;
            }
            Console.WriteLine("Press Enter Key to Exit..");
            Console.ReadLine();
        }
    }
}
  
```

If you observe the above example, we defined a switch with multiple case statements, and it will execute the matched case statements with the expression value.

When we run the above c# program, we will get the result below.



If you observe the above result, the case statement (**20**) matches the defined expression value (**20**) and executes the respective case statement statements.

C# Nested Switch Case Statements

In c#, using one **switch** statement within another **switch** statement is called a **nested switch-case** statement.

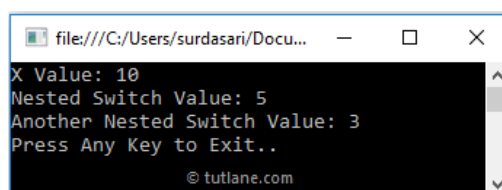
Following is the example of using nested switch statements in the c# programming language.

```
using System;

namespace Tutlane
{
    class Program
    {
        static void Main(string[] args)
        {
            int x = 10, y = 5;
            switch (x)
            {
                case 10:
                    Console.WriteLine("X Value: 10");
                    switch(y){
                        case 5:
                            Console.WriteLine("Nested Switch Value: 5");
                            switch (y - 2) {
                                case 3:
                                    Console.WriteLine("Another Nested Switch Value: 3");
                                    break;
                                }
                            break;
                        }
                    break;
                case 15:
                    Console.WriteLine("X Value: 15");
                    break;
                case 20:
                    Console.WriteLine("X Value: 20");
                    break;
                default:
                    Console.WriteLine("Not Known");
                    break;
            }
            Console.WriteLine("Press Enter Key to Exit..");
            Console.ReadLine();
        }
    }
}
```

If you observe the above example, we used **switch** statements within another **switch** statement to implement nested switch statements based on our requirements.

When we execute the above c# program, we will get the result below.



If you observe the above result, the nested switch statements have been executed based on our requirements.

C# Switch Case Statement with Enum

In c#, we can use **enum** values with Switch case statements to perform required operations.

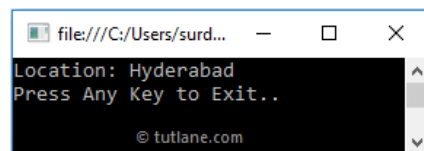
Following is the example of using **enum** values in the c# **switch case** statement.

```
using System;

namespace Tutlane
{
    class Program
    {
        static void Main(string[] args)
        {
            location loc = location.hyderabad;
            switch (loc)
            {
                case location.chennai:
                    Console.WriteLine("Location: Chennai");
                    break;
                case location.guntur:
                    Console.WriteLine("Location: Guntur");
                    break;
                case location.hyderabad:
                    Console.WriteLine("Location: Hyderabad");
                    break;
                default:
                    Console.WriteLine("Not Known");
                    break;
            }
            Console.WriteLine("Press Enter Key to Exit..");
            Console.ReadLine();
        }
        public enum location
        {
            hyderabad,
            chennai,
            guntur
        }
    }
}
```

If you observe the above example, we defined enum values and used those values in switch-case statements to perform required operations based on our requirements.

When we execute the above c# program, we will get the result below.



If you observe the above result, the switch case statement which matches the enum value has been printed in the console window.

This is how we can use enums with switch-case statements to perform operations based on our requirements.

CONTACT US

📍 **Address:** No.1-93, Pochamma Colony, Manikonda, Hyderabad, Telangana - 500089

✉ **Email:** support@tutlane.com (mailto:support@tutlane.com)