**C# switch Statement**

In this article, we will learn about switch statement in C# and how to use them with examples.

Switch statement can be used to replace the [if...else if statement](https://www.programiz.com/csharp-programming/if-else-statement#if-else-if) in C#. The advantage of using switch over if...else if statement is the codes will look much cleaner and readable with switch.

The syntax of switch statement is:

switch (variable/expression)

{

case value1:

// Statements executed if expression(or variable) = value1

break;

case value2:

// Statements executed if expression(or variable) = value1

break;

... ... ...

... ... ...

default:

// Statements executed if no case matches

}

The switch statement evaluates the expression (or variable) and compare its value with the values (or expression) of each case (value1, value2, …). When it finds the matching value, the statements inside that case are executed.

But, if none of the above cases matches the expression, the statements inside default block is executed. The default statement at the end of switch is similar to the else block in if else statement.

However, a problem with the switch statement is, when the matching value is found, it executes all statements after it until the end of switch block.

To avoid this, we use break statement at the end of each case. The break statement stops the program from executing non-matching statements by terminating the execution of switch statement.

To learn more about break statement, visit *C# break statement*.

**Example 1: C# switch Statement**

using System;

namespace Conditional

{

class SwitchCase

{

public static void Main(string[] args)

{

char ch;

Console.WriteLine("Enter an alphabet");

ch = Convert.ToChar(Console.ReadLine());

switch(Char.ToLower(ch))

{

case 'a':

Console.WriteLine("Vowel");

break;

case 'e':

Console.WriteLine("Vowel");

break;

case 'i':

Console.WriteLine("Vowel");

break;

case 'o':

Console.WriteLine("Vowel");

break;

case 'u':

Console.WriteLine("Vowel");

break;

default:

Console.WriteLine("Not a vowel");

break;

}

}

}

}

When we run the program, the output will be:

Enter an alphabet

X

Not a vowel

In this example, the user is prompted to enter an alphabet. The alphabet is converted to lowercase by using ToLower() method if it is in uppercase.

Then, the switch statement checks whether the alphabet entered by user is any of a, e, i, o or u.

If one of the case matches, Vowel is printed otherwise the control goes to default block and Not a vowel is printed as output.

Since, the output for all vowels are the same, we can join the cases as:

**Example 2: C# switch Statement with grouped cases**

using System;

namespace Conditional

{

class SwitchCase

{

public static void Main(string[] args)

{

char ch;

Console.WriteLine("Enter an alphabet");

ch = Convert.ToChar(Console.ReadLine());

switch(Char.ToLower(ch))

{

case 'a':

case 'e':

case 'i':

case 'o':

case 'u':

Console.WriteLine("Vowel");

break;

default:

Console.WriteLine("Not a vowel");

break;

}

}

}

}

The output of both programs is same. In the above program, all vowels print the output Vowel and breaks from the switch statement.

Although switch statement makes the code look cleaner than if...else if statement, switch is restricted to work with limited data types. Switch statement in C# only works with:

* Primitive data types: bool, char and integral type
* *Enumerated Types (Enum)*
* String Class
* Nullable types of above data types

**Example 3: Simple calculator program using C# switch Statement**

using System;

namespace Conditional

{

class SwitchCase

{

public static void Main(string[] args)

{

char op;

double first, second, result;

Console.Write("Enter first number: ");

first = Convert.ToDouble(Console.ReadLine());

Console.Write("Enter second number: ");

second = Convert.ToDouble(Console.ReadLine());

Console.Write("Enter operator (+, -, \*, /): ");

op = (char)Console.Read();

switch(op)

{

case '+':

result = first + second;

Console.WriteLine("{0} + {1} = {2}", first, second, result);

break;

case '-':

result = first - second;

Console.WriteLine("{0} - {1} = {2}", first, second, result);

break;

case '\*':

result = first \* second;

Console.WriteLine("{0} \* {1} = {2}", first, second, result);

break;

case '/':

result = first / second;

Console.WriteLine("{0} / {1} = {2}", first, second, result);

break;

default:

Console.WriteLine("Invalid Operator");

break;

}

}

}

}

When we run the program, the output will be:

Enter first number: -13.11

Enter second number: 2.41

Enter operator (+, -, \*, /): \*

-13.11 \* 2.41 = -31.5951

The above program takes two operands and an operator as input from the user and performs the operation based on the operator.

The inputs are taken from the user using the ReadLine() and Read() method. To learn more, visit [C# Basic Input and Output](https://www.programiz.com/csharp-programming/basic-input-output).

The program uses switch case statement for decision making. Alternatively, we can use if-else if ladder to perform the same operation.