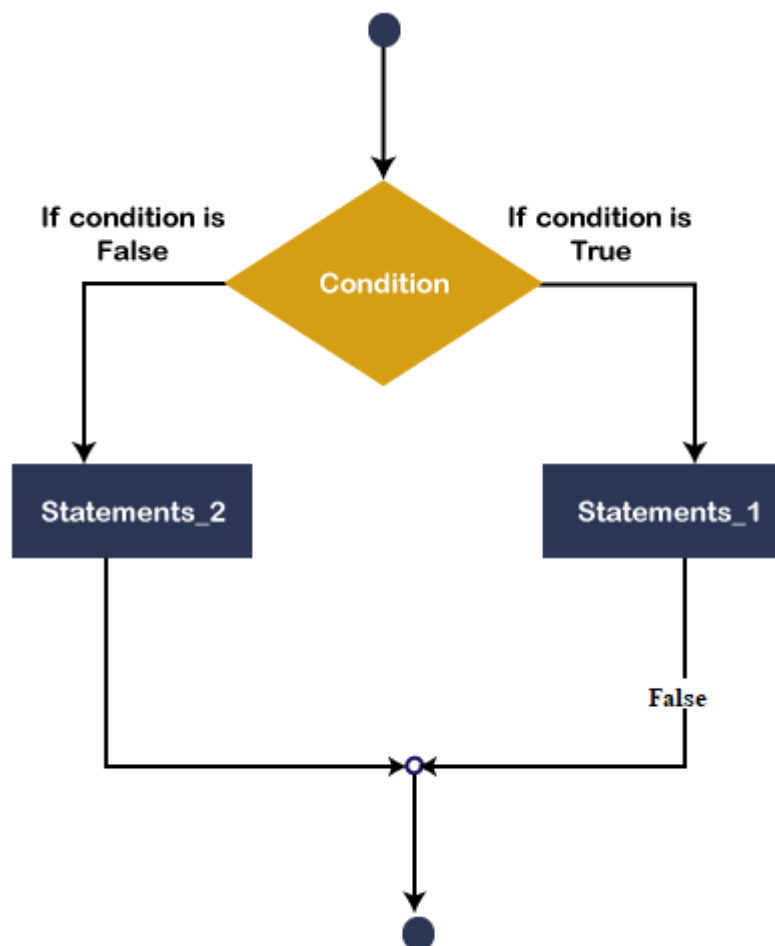


VB.NET Control Statements

In **VB.NET**, the **control statements** are the statements that controls the execution of the program on the basis of the specified condition. It is useful for determining whether a condition is true or not. If the condition is true, a single or block of statement is executed. In the control statement, we will use **if- Then, if Then Else, if Then ElseIf** and the **Select case** statement.

We can define more than one condition to be evaluated by the program with statements. If the defined condition is true, the statement or block executes according to the condition, and if the condition is false, another statement is executed.

The following figure shows a common format of the decision control statements to validate and execute a statement:



The above diagram shows that if the defined condition is true, statement_1 will be executed, and if the condition is false, statement_2 will be executed.

VB.NET provides the following conditional or decision-making statements.

- If-Then Statement
- If-Then Else Statement
- If-Then ElseIf Statement
- Select Case Statement
- Nested Select Case Statements

If-Then Statement

The **If-Then** Statement is a control statement that defines one or more conditions, and if the particular condition is satisfied, it executes a piece of information or statements.

Syntax:

1. If condition Then
2. [Statement or block of Statement]
3. End If

In **If-Then** Statement, the **condition** can be a Boolean, logical, or relational condition, and the statement can be single or group of statements that will be executed when the condition is true.

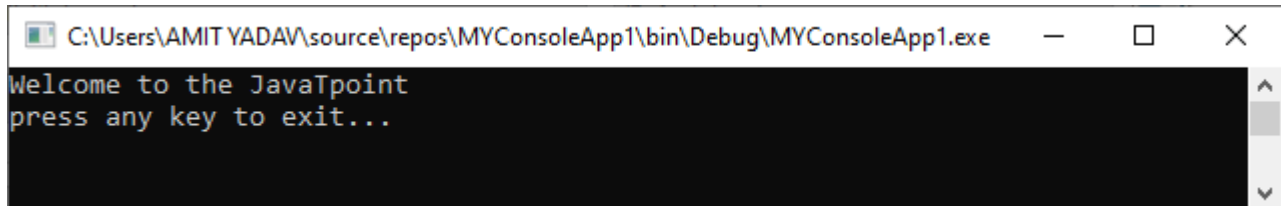
Example 1: Write a simple program to print a statement in VB.NET.

Module1.vb

1. Module Module1
2. ' Declaration of variable str
3. Dim str As String = "JavaTpoint"
4. Sub Main()
5. ' if str equal to "JavaTpoint", below Statement will be executed.
6. If str = "JavaTpoint" Then
7. Console.WriteLine("Welcome to the JavaTpoint")
8. End If
9. Console.WritLine("press any key to exit?")

10. Console.ReadKey()
11. End Sub
12. End Module

Now compile and execute the above program by clicking on the Start or F5 button, it shows the following output:



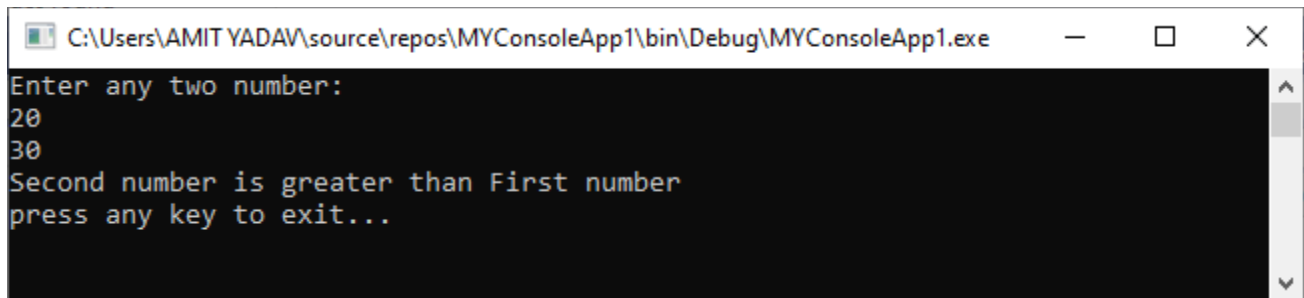
As we can see in the above example, if the value of **str** is equal to **JavaTpoint**, the condition is **true**, and it prints the Statement.

Example 2: Write a program to print a number is greater than another number in VB.NET.

if_statment2.vb

1. Module if_statement2
2. Sub Main()
3. ?Definition of variables
4. Dim no1, no2 As Integer
5. Console.WriteLine("Enter any two number:")
6. no1 = Console.ReadLine() ?read no1 from user
7. no2 = Console.ReadLine() ?read no2 from user
8. If no1 > no2 Then
9. Console.WriteLine("First number is greater than second number")
10. End If
11. If no1 < no2 Then
12. Console.WriteLine("Second number is greater than First number")
13. End If
14. Console.WriteLine("press any key to exit...")
15. Console.ReadKey()
16. End Sub
17. End Module

Now compile and execute the above program by clicking on the Start or F5 button, it shows the following output:



```
C:\Users\AMIT YADAV\source\repos\MYConsoleApp1\bin\Debug\MYConsoleApp1.exe
Enter any two number:
20
30
Second number is greater than First number
press any key to exit...
```

In the above program, we enter two numbers to find the greater number using the relational operator. And if the first number is greater than the other, the first statement is executed; otherwise, the second statement will be executed.

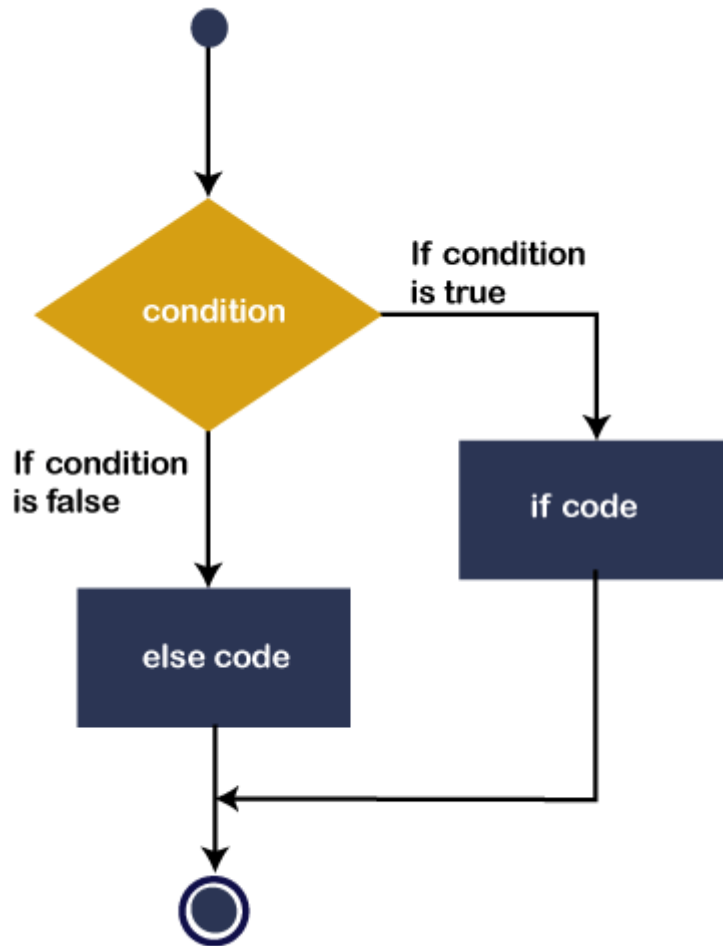
If-Then-Else Statement

The **If-Then** Statement can execute single or multiple statements when the condition is true, but when the expression evaluates to **false**, it does nothing. So, here comes the **If-Then-Else** Statement. The IF-Then-Else Statement is telling what **If** condition to do when if the statement is false, it executes the Else statement. Following is the If-Then-Else statement syntax in VB.NET as follows:

Syntax:

1. If (Boolean_expression) Then
2. 'This statement will execute **if** the Boolean condition is **true**
3. Else
4. 'Optional statement will execute **if** the Boolean condition is **false**
5. End If

Flow chart



The above diagram represents that if the Boolean expression (condition) is **true**, the if statement will execute, and if the Boolean expression is false, **Else code or statement** will be executed. After that, the control transfer to the next statement, which is immediately after the If-Then-Else control statement.

Example 1: Write a program to check whether the number is even or odd.

If_Else_statment.vb

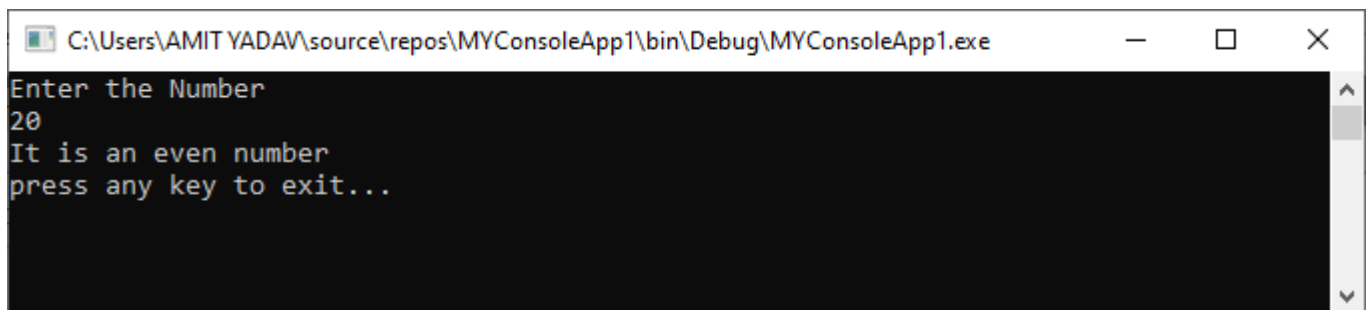
1. Module If_Else_statement
2. Sub Main()
3. Dim num As Integer
4. Console.WriteLine("Enter the Number")
5. num = Console.ReadLine() 'read data from console
- 6.
7. If (num Mod 2 = 0) Then ' **if** condition is **true**, print the **if** statement

```

8.      Console.WriteLine("It is an even number")
9.
10.     Else 'otherwise, Else statement is executed.
11.      Console.WriteLine("It is an odd number")
12.     End If
13.
14.     Console.WriteLine("press any key to exit...")
15.     Console.ReadKey()
16. End Sub
17. End Module

```

Now compile and execute the above program by clicking on the Start or F5 button, it shows the following output:



```

C:\Users\AMIT YADAV\source\repos\MYConsoleApp1\bin\Debug\MYConsoleApp1.exe
Enter the Number
20
It is an even number
press any key to exit...

```

Example 2: Write a program to print the larger and smaller of the two numbers.

if_else_statment2.vb

```

1. Module if_else_statement2
2.     Sub Main()
3.         Dim a As Integer
4.         Dim b As Integer
5.         Console.WriteLine("Enter the first number : ")
6.         a = Console.ReadLine()
7.
8.         Console.WriteLine("Enter the second number : ")
9.         b = Console.ReadLine()
10.
11.        If a > b Then
12.            Console.WriteLine(" larger number = {0} and smaller number = {1} ", a, b
13.        )
14.        Else

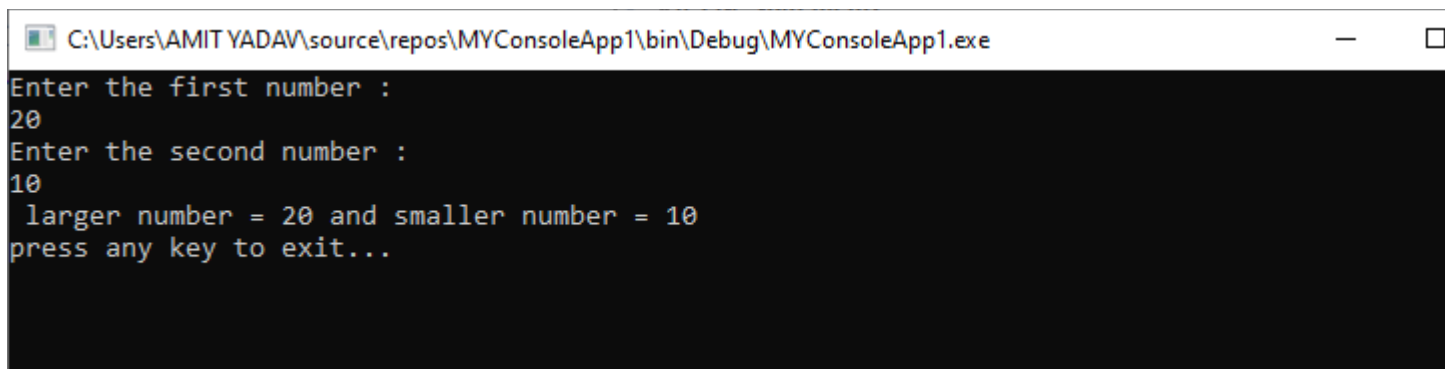
```

```

14.      Console.WriteLine(" larger number = {0} and smaller number = {1} ", b, a
)
15.      End If
16.
17.      Console.WriteLine("press any key to exit...")
18.      Console.ReadKey()
19.  End Sub
20.End Module

```

Now compile and execute the above program by clicking on the Start or F5 button, it shows the following output:



```

C:\Users\AMITYADAV\source\repos\MYConsoleApp1\bin\Debug\MYConsoleApp1.exe
Enter the first number :
20
Enter the second number :
10
larger number = 20 and smaller number = 10
press any key to exit...

```

VB.NET If-Then-ElseIf statement

The **If-Then-ElseIf** Statement provides a choice to execute only one condition or statement from multiple statements. Execution starts from the top to bottom, and it checked for each If condition. And if the condition is met, the block of If the statement is executed. And **if none** of the conditions are true, the last **block** is executed. Following is the syntax of If-Then-ElseIf Statement in VB.NET as follows:

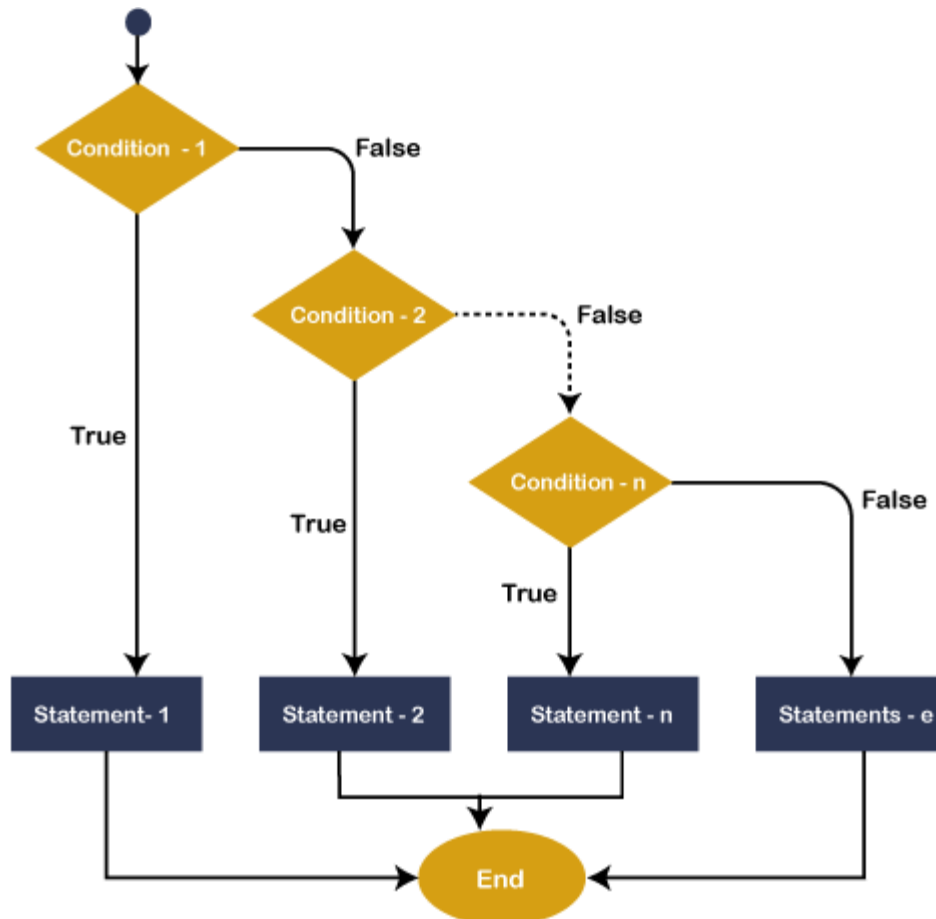
Syntax

1. If(condition 1)Then
2. ' Executes when condition 1 is **true**
3. ElseIf(condition 2)Then
4. ' Executes when condition 2 is **true**
5. ElseIf(boolean_expression 3)Then
6. ' Executes when the condition 3 is **true**
7. Else
8. ' executes the **default** statement when none of the above conditions is **true**.

9. End If

Flowchart

The following diagram represents the functioning of the If-Else-If Statement in the VB.NET programming language.



If this condition is true in the flowchart of the if-else-if statement, the statement is executed within the if block. If the condition is not true, it passes control to the next ElseIf condition to check whether the condition is matched. And if none of the conditions are matched, the else block is executed.

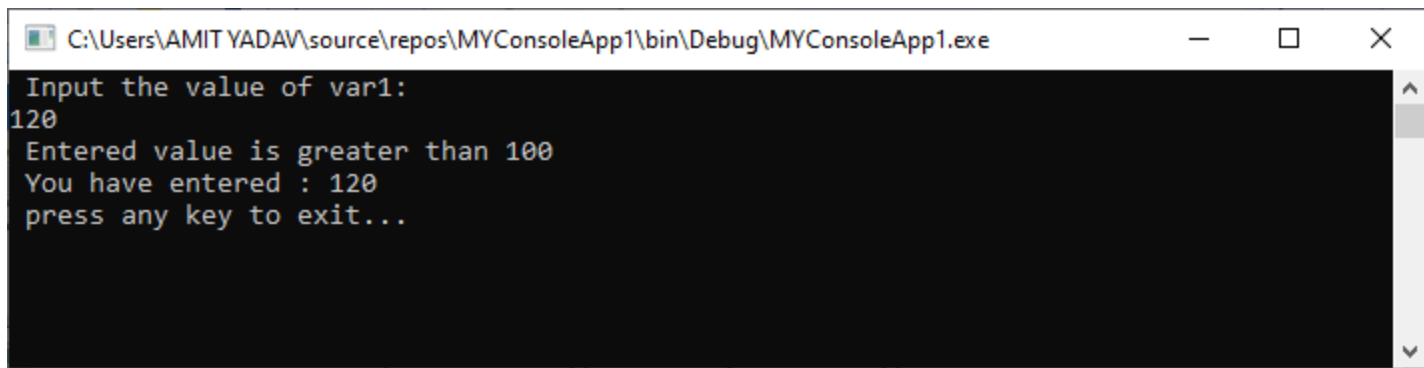
Example 1: Write a program to show the uses of If... ElseIf statements.

if_elseIf.vb

1. Module if_elseIf
2. Sub Main()


```
3. Dim var1 As Integer
4.
5. Console.WriteLine(" Input the value of var1: ")
6. var1 = Console.ReadLine()
7. If var1 = 20 Then
8.     'if condition is true then print the following statement'
9.     Console.WriteLine(" Entered value is equal to 20")
10. ElseIf var1 < 50 Then
11.     Console.WriteLine(" Entered value is less than 50")
12.
13. ElseIf var1 >= 100 Then
14.     Console.WriteLine(" Entered value is greater than 100")
15. Else
16.     'if none of the above condition is satisfied, print the following statement
17.     Console.WriteLine(" Value is not matched with above condition")
18. End If
19. Console.WriteLine(" You have entered : {0}", var1)
20. Console.WriteLine(" press any key to exit...")
21. Console.ReadKey()
22. End Sub
23. End Module
```

Now compile and execute the above program by clicking on the Start or F5 button, it shows the following output:



```
C:\Users\AMIT YADAV\source\repos\MYConsoleApp1\bin\Debug\MYConsoleApp1.exe
Input the value of var1:
120
Entered value is greater than 100
You have entered : 120
press any key to exit...
```

Example 2: Write a program to use the If-Then-ElseIf Statement for calculating the division obtained by the student. Also, take the marks obtained by the student in 5 different subjects from the keyboard.

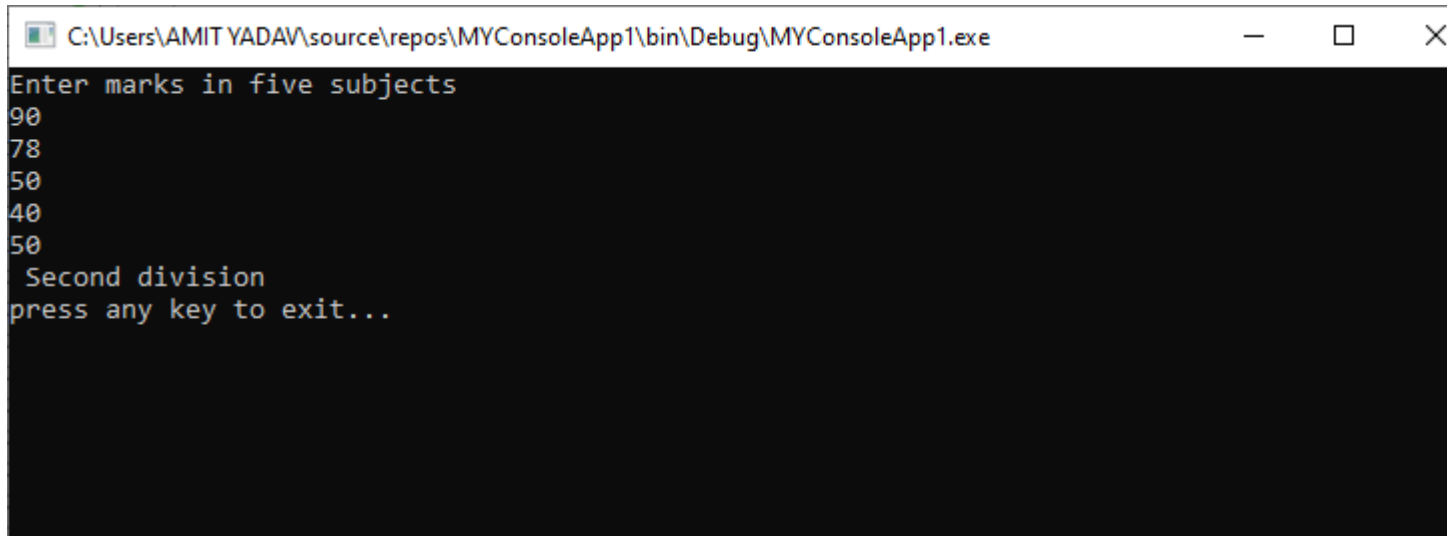
if_elseIf2.vb

```

1. Module If_elseIf2
2.   Sub Main()      ' execution start from Main() method
3.       Dim m1, m2, m3, m4, m5, per As Integer
4.       Console.WriteLine("Enter marks in five subjects ")
5.       ' Read the marks of five subject
6.       m1 = Console.ReadLine()
7.       m2 = Console.ReadLine()
8.       m3 = Console.ReadLine()
9.       m4 = Console.ReadLine()
10.      m5 = Console.ReadLine()
11.      per = (m1 + m2 + m3 + m4 + m5) / 5
12.      If (per >= 70) Then
13.          'if condition is true, print the first division
14.          Console.WriteLine(" First division")
15.      ElseIf (per >= 60) Then
16.          'if ElseIf condition is true, print the second division
17.          Console.WriteLine(" Second division")
18.      ElseIf (per >= 50) Then
19.          'if ElseIf condition is true, print the third division
20.          Console.WriteLine(" Third division")
21.      ElseIf (per >= 40) Then
22.          'if ElseIf condition is true, print only pass with grace
23.          Console.WriteLine(" Only Pass with Grace")
24.      Else
25.          'if none of the condition is true, print the Failed
26.          Console.WriteLine(" Failed")
27.      End If
28.      Console.WriteLine("press any key to exit...")
29.      Console.ReadKey()
30.  End Sub
31.
32.End Module

```

Now compile and execute the above program by clicking on the Start or F5 button, it shows the following output:



```
C:\Users\AMIT YADAV\source\repos\MYConsoleApp1\bin\Debug\MYConsoleApp1.exe
Enter marks in five subjects
90
78
50
40
50
Second division
press any key to exit...
```

Select Case Statement

In VB.NET, the Select Case statement is a collection of multiple case statements, which allows executing a single case statement from the list of statements. A selected case statement uses a variable to test for equality against multiple cases or statements in a program. If the variable is matched with any test cases, that statement will be executed. And if the condition is not matched with any cases, it executes the default statement.

Using the select case statement in VB.NET programming, you can replace the uses of multiple If-Then-Else If statement from the program for better readability and easy to use.

Syntax

Following is the syntax of the Select Case statement in VB.NET, as follows:

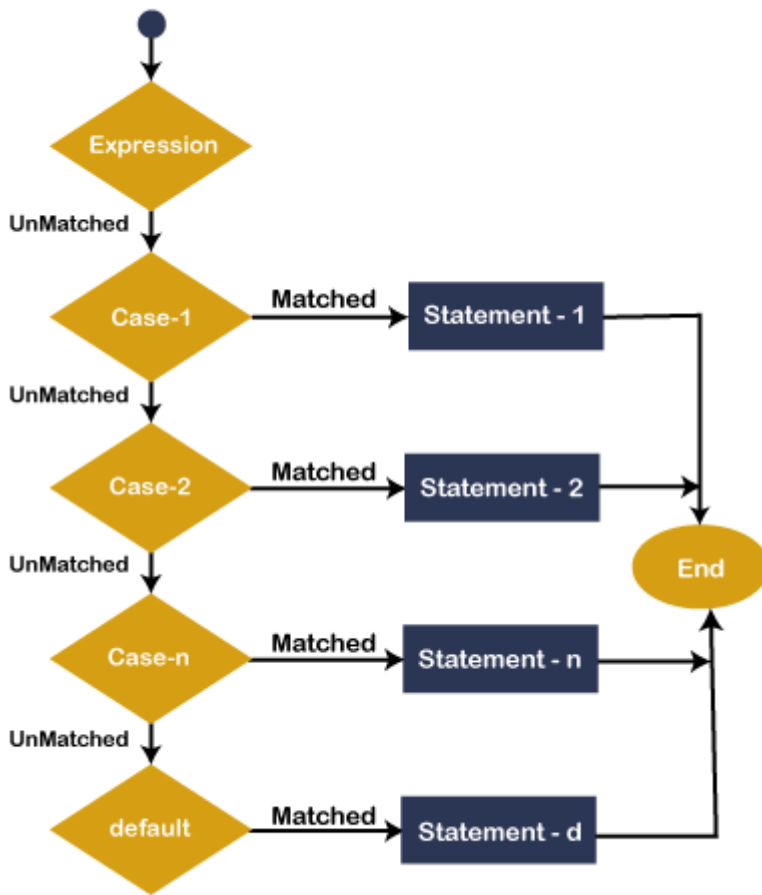
1. Select Case [variable or expression]
2. Case value1 'defines the item or value that you want to match.
3. // Define a statement to execute
- 4.
5. Case value2 'defines the item or value that you want to match.
6. // Define a statement to execute
- 7.
8. Case Else
9. // Define the default statement if none of the conditions is true.
10. End Select

Furthermore, you can also set more than one condition in a single case statement, such as:

1. Select Case Variable / expression
2. Case value1
3. Statement1
- 4.
5. Case value2, value3
6. Statement2
- 7.
8. Case Else
9. // define the default statement if none of the condition is true
10. End Select

Flowchart of Select Case Statement

The following flowchart represents the functioning of the Select case statement in the VB.NET programming language.



In Flowchart, the Select Case statement represents the evaluating of the process start from top to bottom. If the expression or value is matched with the first select case, statement - 1 is executed else the control transfer to the next case for checking whether the expression is matching or not. Similarly, it checks all Select case statements for evaluating. If none of the cases are matched, the Else block statement will be executed, and finally, the Select Case Statement will come to an end.

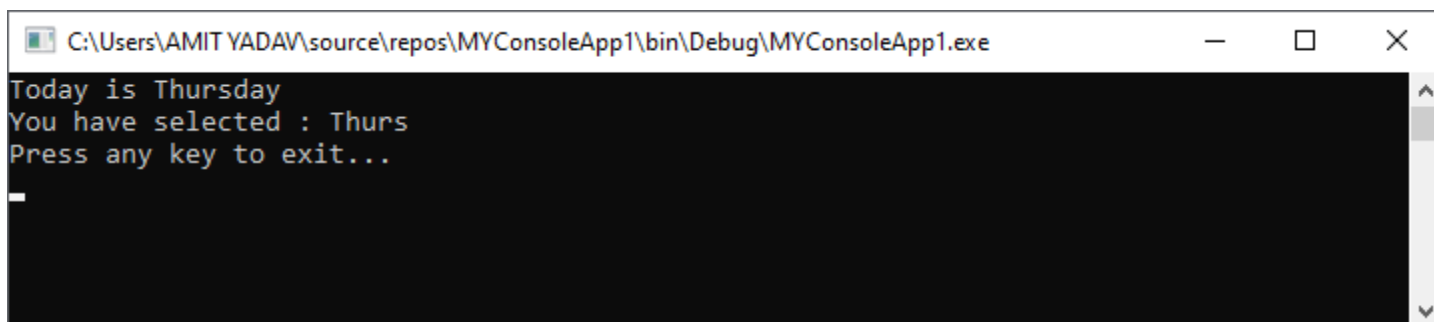
Example 1: Write a program to display the Days name using the select case statement in VB.NET.

Select_case.vb

1. Imports System
2. Module Select_case
3. Sub Main()

```
4.      'define a local variable.
5.      Dim Days As String
6.      Days = "Thurs"
7.      Select Case Days
8.          Case "Mon"
9.              Console.WriteLine(" Today is Monday")
10.         Case "Tue"
11.             Console.WriteLine(" Today is Tuesday")
12.         Case "Wed"
13.             Console.WriteLine("Today is Wednesday")
14.         Case "Thurs"
15.             Console.WriteLine("Today is Thursday")
16.         Case "Fri"
17.             Console.WriteLine("Today is Friday")
18.         Case "Sat"
19.             Console.WriteLine("Today is Saturday")
20.         Case "Sun"
21.             Console.WriteLine("Today is Sunday")
22.         Case Else
23.             Console.WriteLine(" You have typed Something wrong")
24.
25.     End Select
26.     Console.WriteLine("You have selected : {0}", Days)
27.     Console.WriteLine("Press any key to exit...")
28.     Console.ReadLine()
29. End Sub
30. End Module
```

Now compile and execute the above program by clicking on the Start or F5 button, it shows the following output:



The screenshot shows a Windows console window titled "C:\Users\AMITYADAV\source\repos\MYConsoleApp1\bin\Debug\MYConsoleApp1.exe". The console output is as follows:

```
Today is Thursday
You have selected : Thurs
Press any key to exit...
_
```

In the select case statement, the value of Days "**Thurs**" will compare all the available **select cases**' values in a program. If a value matched with any condition, it prints the particular statement, and if the value is not matched with any select case statement, it prints the default message.

Example 2: Write a program to perform an arithmetic operation using the Select case statement in VB.NET.

Operation.vb

Operation.vb

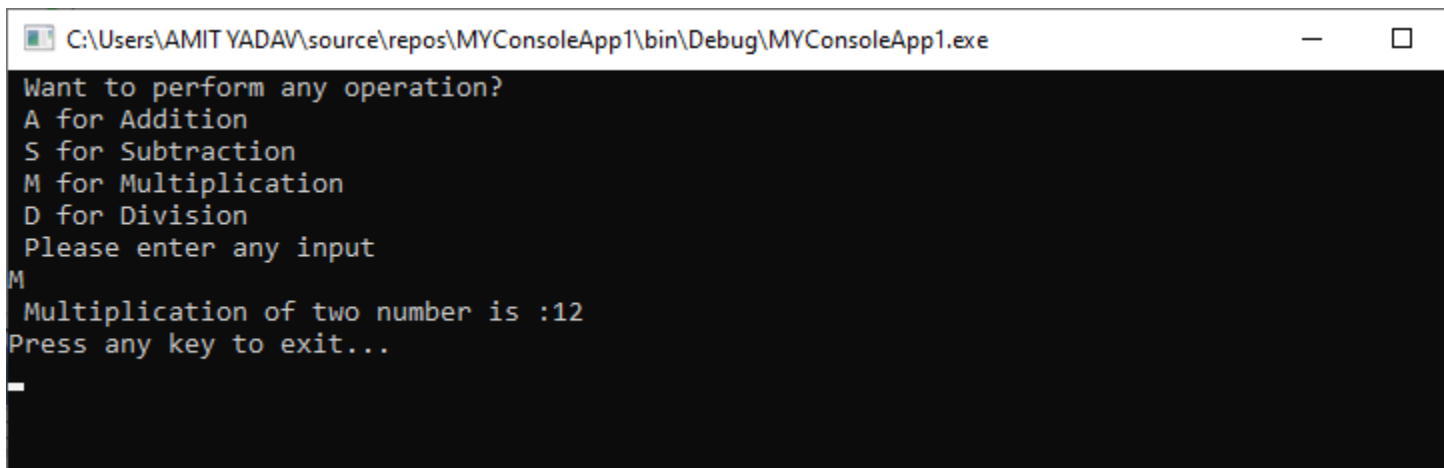
```
1. Imports System
2. Module Operation
3.     Sub main()
4.         'declaration of the variables
5.         Dim num1, num2, sum As Integer
6.         Dim def As Char
7.         'initialization of num1 and num2 variable
8.         num1 = 2
9.         num2 = 6
10.        Console.WriteLine(" Want to perform any operation?")
11.        Console.WriteLine(" A for Addition")
12.        Console.WriteLine(" S for Subtraction")
13.        Console.WriteLine(" M for Multiplication")
14.        Console.WriteLine(" D for Division")
15.        Console.WriteLine(" Please enter any input")
16.        def = Console.ReadLine()
17.        Select Case def
18.            Case "A"
19.                'perform Addition
20.                sum = num1 + num2
21.                Console.WriteLine(" Addition of two number is :{0}", sum)
22.            Case "S"
23.                'perform Subtraction
24.                sum = num2 - num1
25.                Console.WriteLine(" Subtraction of two number is :{0}", sum)
26.            Case "M"
27.                'perform Multiplication
28.                sum = num1 * num2
```

```

29.      Console.WriteLine(" Multiplication of two number is :{0}", sum)
30.      Case "D"
31.          'Peform Division
32.          sum = num2 / num1
33.          Console.WriteLine(" Division of two number is :{0}", sum)
34.      Case Else
35.          'If none of the operation matched, call default statement
36.          Console.WriteLine(" Please enter only define operation With Capital lett
er")
37.      End Select
38.      Console.WriteLine("Press any key to exit...")
39.      Console.ReadKey()
40.  End Sub
41.End Module

```

Now compile and execute the above program by clicking on the Start or F5 button, it shows the following output:



```

C:\Users\AMITYADAV\source\repos\MYConsoleApp1\bin\Debug\MYConsoleApp1.exe
Want to perform any operation?
A for Addition
S for Subtraction
M for Multiplication
D for Division
Please enter any input
M
Multiplication of two number is :12
Press any key to exit...

```

In the above example, we defined Select with multiple case statements, and if the **user-defined** input is matched with any defined case statement, it executes that statement. And if the condition is not matched with any case, it executes a default statement in VB.NET.

Here, we provide 'M' as input, which checks all case statements, and if any case is matched with M, it executes the statement within the respective Case statement.

VB.NET Nested Select Case statements

When a **Select Case** statement is written inside the body of another **Select Case statement** is called a **nested Select Case statement**.

Syntax:

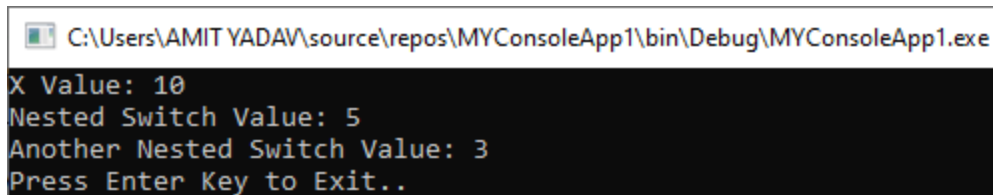
```
1. Select Case "num"
2. ' code to be executed if num = 1
3. Case 1
4. ' nested Select case
5.     Select Case n
6.
7.         ' code to be executed if n = 5
8.         Case 5
9.         Statement 1
10.
11.        ' code to be executed if n = 10
12.        Case 10
13.        Statement 2
14.
15.        ' code to be executed if n = 15
16.        Case 15
17.        Statement 3
18.
19.        ' code to be executed if n doesn't match with any cases.
20.        Case Else
21.        Statement
22.
23.    ' code to be executed if num = 2
24.    Case 2
25.    Statement 2
26.
27.    ' code to be executed if num = 3
28.    Case 3
29.    Statement 3
30.
31.    ' code to be executed if num doesn't match with any cases.
32.    Case Else
33.    Statement
```

Example 1: Write a program to use a nested select case statement in VB.NET.

Module1.vb

```
1. Module Module1
2.
3.     Sub Main()
4.
5.         Dim x As Integer = 10, y As Integer = 5
6.         Select Case x
7.
8.             Case 10
9.                 Console.WriteLine("X Value: 10")
10.
11.
12.             Select Case y
13.                 Case 5
14.                     Console.WriteLine("Nested Switch Value: 5")
15.
16.
17.                 Select Case y - 2
18.                     Case 3
19.                         Console.WriteLine("Another Nested Switch Value: 3")
20.
21.                 End Select
22.             End Select
23.
24.             Case 15
25.                 Console.WriteLine("X Value: 15")
26.
27.             Case 20
28.                 Console.WriteLine("X Value: 20")
29.
30.             Case Else
31.                 Console.WriteLine("Not Known")
32.
33.         End Select
34.         Console.WriteLine("Press Enter Key to Exit..")
35.         Console.ReadLine()
36.     End Sub
37. End Module
```

Now compile and execute the above program by clicking on the Start or F5 button, it shows the following output:



A screenshot of a Windows console application window. The title bar shows the file path: C:\Users\AMIT YADAV\source\repos\MYConsoleApp1\bin\Debug\MYConsoleApp1.exe. The console output is as follows:

```
X Value: 10
Nested Switch Value: 5
Another Nested Switch Value: 3
Press Enter Key to Exit..
```

Example 2: Write a program to use the nested select case statement in VB.NET.

nested_selectcase.vb

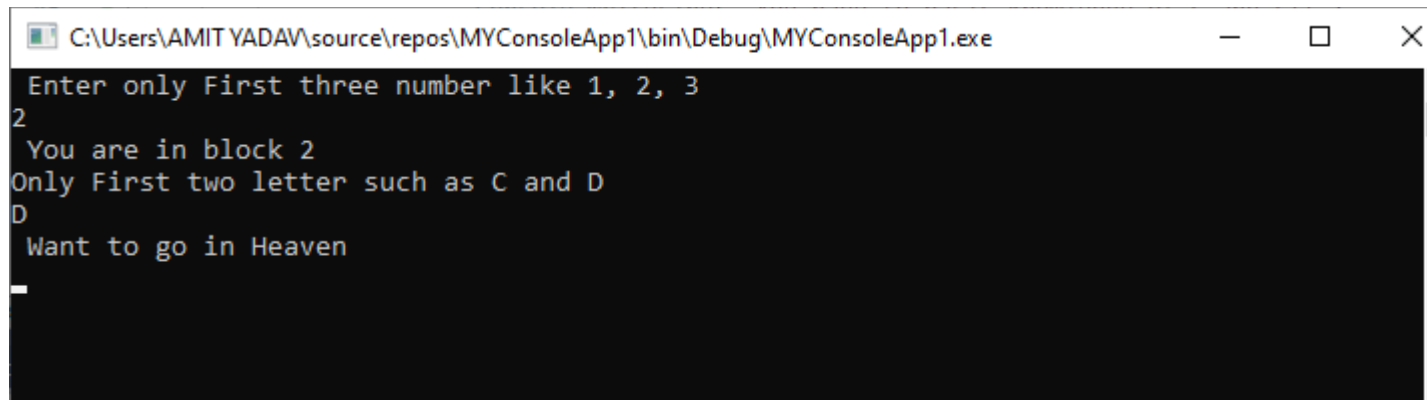
```
1. Imports System
2. Module nested_selectcase
3.     Sub Main()
4.         Dim num As Integer
5.         Dim str As String
6.         str = "F"
7.
8.         Console.WriteLine(" Enter only First three number like 1, 2, 3")
9.         num = Console.ReadLine() 'take input from the user
10.        Select Case num
11.            Case 1
12.                Console.WriteLine(" You are in block 1")
13.                Console.WriteLine("Only First two letter such as A and B")
14.
15.                str = Console.ReadLine()
16.                Select Case str
17.                    Case "A", "a"
18.                        Console.WriteLine(" This is a VB.NET Tutorial")
19.                    Case "B", "b"
20.                        Console.WriteLine(" Welcome to the JavaTpoint")
21.                    Case Else
22.                        Console.WriteLine(" Something is wrong")
23.                End Select
24.
```

```

25. Case 2
26. Console.WriteLine(" You are in block 2")
27. Console.WriteLine("Only First two letter such as C and D")
28. str = Console.ReadLine()
29. Select Case str
30. Case "C", "c"
31. Console.WriteLine(" Welcome to the World!")
32. Case "D", "d"
33. Console.WriteLine(" Want to go in Heaven")
34. Case Else
35. Console.WriteLine(" Something is wrong")
36. End Select
37.
38. Case 3
39. Console.WriteLine(" You are in block 3")
40. Console.WriteLine("Only First two letter such as E and F")
41. str = Console.ReadLine()
42. Select Case str
43. Case "E", "e"
44. Console.WriteLine(" VB.NET is a programming language to develop web, window, and console-based application. ")
45. Case "F", "f"
46. Console.WriteLine(" You have to basic knowledge of c and c++")
47. Case Else
48. Console.WriteLine(" Something is wrong")
49. End Select
50. Case Else
51. Console.WriteLine(" Something is wrong")
52. End Select
53. Console.ReadLine()
54. End Sub
55. End Module

```

Now compile and execute the above program by clicking on the Start or F5 button, it shows the following output:

A screenshot of a Windows console window titled "C:\Users\AMIT YADAV\source\repos\MYConsoleApp1\bin\Debug\MYConsoleApp1.exe". The console has a black background with white text. The text displayed is: "Enter only First three number like 1, 2, 3", followed by the user input "2". Then it says "You are in block 2", followed by the user input "D". Finally, it displays "Only First two letter such as C and D" and "Want to go in Heaven". A white cursor is visible on the line following "Want to go in Heaven".

```
C:\Users\AMIT YADAV\source\repos\MYConsoleApp1\bin\Debug\MYConsoleApp1.exe
Enter only First three number like 1, 2, 3
2
You are in block 2
Only First two letter such as C and D
D
Want to go in Heaven
_
```

In the above example, we have only defined the first three numbers 1-3 and if the number matches to any case statement, the select statement is executed. Here, we have entered 2 that is matched with case 2 and it executes a block as shown above. And this block executes the statement "Only the First two letters such as C and D". Therefore, we enter a letter D, letter D is matched with the nested select case statement, and if a match is found, it executes the select case statement as shown above.