Functions in VB.NET

Kavita K. Bharti
Assistant Professor
Department of Computer
Durga Mahavidyalaya, Raipur

A procedure is a group of statements that together perform a task when called. After the procedure is executed, the control returns to the statement calling the procedure. VB.Net has two types of procedures −

- **Functions**
- Sub procedures or Subs

Functions return a value, whereas Subs do not return a value.

Defining a Function

The Function statement is used to declare the name, parameter and the body of a function. The syntax for the Function statement is −

[Modifiers] Function FunctionName [(ParameterList)] As ReturnType
[Statements]
End Function

Where,

- **Modifiers** − specify the access level of the function; possible values are: Public, Private, Protected, Friend, Protected Friend and information regarding overloading, overriding, sharing, and shadowing.
- **FunctionName** − indicates the name of the function
- **ParameterList** − specifies the list of the parameters
- **ReturnType** − specifies the data type of the variable the function returns

Following code snippet shows a function *FindMax* that takes two integer values and returns the larger of the two.

```
Function Max(ByVal num1 As Integer, ByVal num2 As Integer) As Integer
  Dim result As Integer
    If (num1 > num2) Then
     result = num1
  Else
     result = num2
```

```
  End If
  Max = result
End Function
```

Function Returning a Value

In VB.Net, a function can return a value to the calling code in two ways −

- By using the return statement
- By assigning the value to the function name

The following example demonstrates using the *FindMax* function −

```
Module myfunctions
  Function Max(ByVal num1 As Integer, ByVal num2 As Integer) As Integer
    Dim result As Integer
        If (num1 > num2) Then
      result = num1
    Else
      result = num2
    End If
    FindMax = result
  End Function
  Sub Main()
    Dim a As Integer = 100
    Dim b As Integer = 200
    Dim res As Integer

    res = Max(a, b)
    Console.WriteLine("Max value is : {0}", res)
    Console.ReadLine()
  End Sub
End Module
```

When the above code is compiled and executed, it produces the following result −

Max value is : 200

Recursive Function

A function can call itself. This is known as recursion. Following is an example that calculates factorial for a given number using a recursive function −

```
Module myfunctions
  Function factorial(ByVal num As Integer) As Integer
    Dim result As Integer
     If (num = 1) Then
      Return 1
    Else
      result = factorial(num - 1) * num
      Return result
    End If
  End Function
  Sub Main()
    'calling the factorial method
    Console.WriteLine("Factorial of 6 is : {0}", factorial(6))
    Console.WriteLine("Factorial of 7 is : {0}", factorial(7))
    Console.WriteLine("Factorial of 8 is : {0}", factorial(8))
    Console.ReadLine()
  End Sub
End Module
```

When the above code is compiled and executed, it produces the following result −

Factorial of 6 is: 720

Factorial of 7 is: 5040

Factorial of 8 is: 40320