Syllabus Content:

8.1. Programming Concepts (Procedures & Functions)

- 6 (a) Understand what is meant by procedures, functions and parameters
- (b) Define and use procedures and functions, with or without parameters
- (c) Understand and use local and global variables.

Notes and guidance

- Procedures and functions may have up to two

8.1. Programming Concept (Procedures & Functions)

Algorithm design involves developing **step-by-step** instructions to solve a problem and subroutines are to **modularize the solution**.

Initially, a program was written as one monolithic block of code. The program started at the first line of the program and continued to the end.

CS (2210) with Sir Majid at www.majidtahir.com

Program languages have now been developed to be structured. A problem can be divided into a number of smaller subroutines (also called **procedures**). From within one subroutine, another subroutine can be called and executed:

Computer Sc.(9618) with Majid Tahir at www.majidt

PROCEDURE or Subroutine:

A **PROCEDURE** is a self-contained section of program code which performs a specific task and is referenced by a name. Procedures can be **CALLED** repeatedly throughout a program by keyword **CALL**. **PROCEDURE** can contain its own local variables, data types, labels, and constant declarations.

FUNCTION: is a self contained program code which **performs a specific task** and is referenced by a name. **FUNCTION always return a value**..FUNCTION is a sequence of steps that is given an identifier and returns a single value; function call is made by the keyword **CALL**

Parameter of Procedure or Function:

A **variable applied to a procedure or function** that allows one to **pass in a value** for the **procedure/function** to **use.**

Header (Procedure or Function): the first statement in the definition of a procedure or function, which contains its name, any parameters passed to it, and, for a function, the type of the return value.

Argument – the value passed to a procedure or function.

Local and global variables

A **global variable** can be used by any part of a program – its **scope** covers the whole program. **Global Variables are DECLARED above and outside** the **Main Program**, **Procedures or Functions**

A **local variable** can only be used by the part of the program it has been **declared inside** – its **scope** is **restricted to that part of the program.**

For example, in this algorithm the variables **num1**, **num2** and **answer** are declared globally.

```
DECLARE num1, num2, answer : Integer

     PROCEDURE userinput()
          OUTPUT("Enter number 1")
          INPUT num1
          OUTPUT("Enter number 2")
          INPUT num2
     END PROCEDURE

     PROCEDURE Calculation()
          answer = num1 * num2
     END PROCEDURE

     PROCEDURE useroutput()
          OUTPUT("the product of ", num1 ," and ", num2 ," is ")
          OUTPUT (CALL answer)
     END PROCEDURE
 BEGIN
   CALL userinput
   CALL Calculation
   CALL useroutput
 END
```

CS (2210) with Sir Majid at www.majidtahir.com
Computer Sc (9618) with Majid Tahir at www.majidtahir.com

Subroutines or Procedures

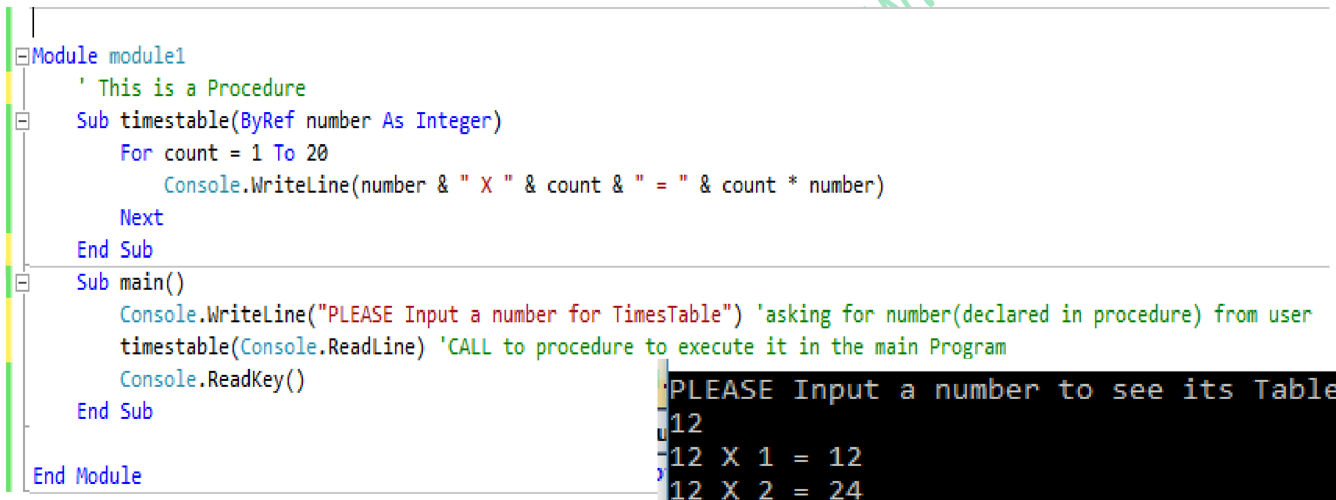
Subroutine

A subroutine is another name of PROCEDURE. It is a self-contained section of program code that performs a specific task, as part of the main program.

PSEUDOCODE of PROCEDURE

```
PROCEDURE timestable(number As INTEGER) //This is a Procedure
   FOR count = 1 To 20
       OUTPUT(number & " X " & count & " = " & count * number)
   NEXT
END PROCEDURE

BEGIN
OUTPUT("PLEASE Input number for TimesTable") //asking for number(in
procedure) from user
CALL timestable //CALL to procedure to execute it in the main Program
END
```

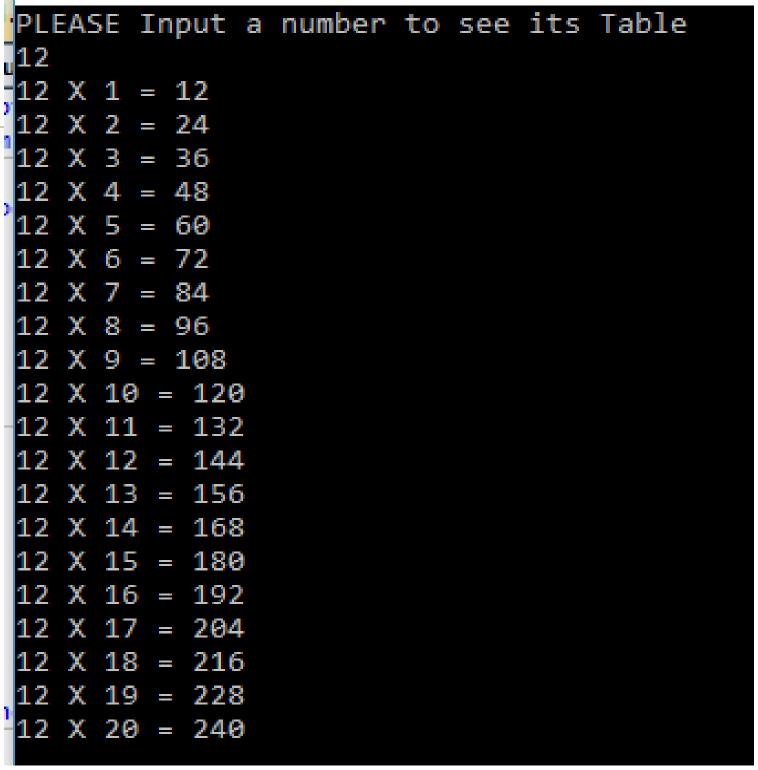
VB CODE Of PROCEDURE



```
Module module1
    ' This is a Procedure
    Sub timestable(ByRef number As Integer)
        For count = 1 To 20
            Console.WriteLine(number & " X " & count & " = " & count * number)
        Next
    End Sub
    Sub main()
        Console.WriteLine("PLEASE Input a number for TimesTable") 'asking for number(declared in procedure) from user
        timestable(Console.ReadLine) 'CALL to procedure to execute it in the main Program
        Console.ReadKey()
    End Sub

End Module
```

Computer Sc. (9618) www.maj

OUTPUT is shown in COLSOLE WINDOW:



```
PLEASE Input a number to see its Table
12
12 X 1 = 12
12 X 2 = 24
12 X 3 = 36
12 X 4 = 48
12 X 5 = 60
12 X 6 = 72
12 X 7 = 84
12 X 8 = 96
12 X 9 = 108
12 X 10 = 120
12 X 11 = 132
12 X 12 = 144
12 X 13 = 156
12 X 14 = 168
12 X 15 = 180
12 X 16 = 192
12 X 17 = 204
12 X 18 = 216
12 X 19 = 228
12 X 20 = 240
```

PROCEDURE of Fahrenheit to Celsius

```
PROCEDURE Celsius (temp : REAL)
     temp = (temp -32)/1.8
     OUTPUT("Celsius = " , temp)
END PROCEDURE
BEGIN
     DECLARE MyTemp : REAL
     OUTPUT("Input temperature in Fahrenheit ")
     INPUT MyTemp
     CALL Celsius(MyTemp) //CALL to procedure to execute
END
```

Using Global variable (Pseudocode)

```
DECLARE num1, num2, answer As Integer
     PROCEDURE input_sub()
          OUTPUT("Enter number 1")
          INPUT num1 = Console.ReadLine
          OUTPUT("Enter number 2")
          INPUT num2
     END PROCEDURE
     PROCEDURE Calculation()
          answer = num1 * num2
     END PROCEDURE
     PROCEDURE output_sub()
          OUTPUT("the product of " & num1 & " and " & num2 & " is ")
          OUTPUT (answer)
     END PROCEDURE
    BEGIN
        CALL  input_sub()
        CALL  Calculation()
        CALL  output_sub()
    END
```

Computer Sc (9618) with Majid Tahir at www.majidtahir.com
CS (2210) with Sir Majid at www.majidtahir.com

Example Program – Procedures (VB Code)

```
Dim num1 As Integer
Dim num2 As Integer
Dim answer As Integer
```

Global Variables declared

```
  Sub input_sub()
        Console.Clear()
        Console.WriteLine("Enter number 1")
        num1 = Console.ReadLine
        Console.WriteLine("Enter number 2")
        num2 = Console.ReadLine
  End Sub
  Sub Calculation()
        answer = num1 * num2
  End Sub
Sub output_sub()
  Console.Write("the product of " & num1 & " and " & num2 & " is ")
  Console.WriteLine(answer)
  Console.ReadLine()
End Sub
Sub Main()
    input_sub()
    Calculation()
    output_sub()
End Sub
```

Computer Sc (0618) with Majid Tahir at www.majidtahir.com
CS (2210) with Sir Majid at www.majidtahir.com

Parameters

As mentioned above, **local variables** only have a **lifespan of the procedure.**
Sometimes it is useful to **pass a value from one procedure to another**.
This is done by using **parameters (or arguments)**
A parameter can be passed from one procedure to another by value or by reference.

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:

```
FUNCTION FunctionName [(ParameterList)] As ReturnType

    [Statements]
```

```
End Function
```

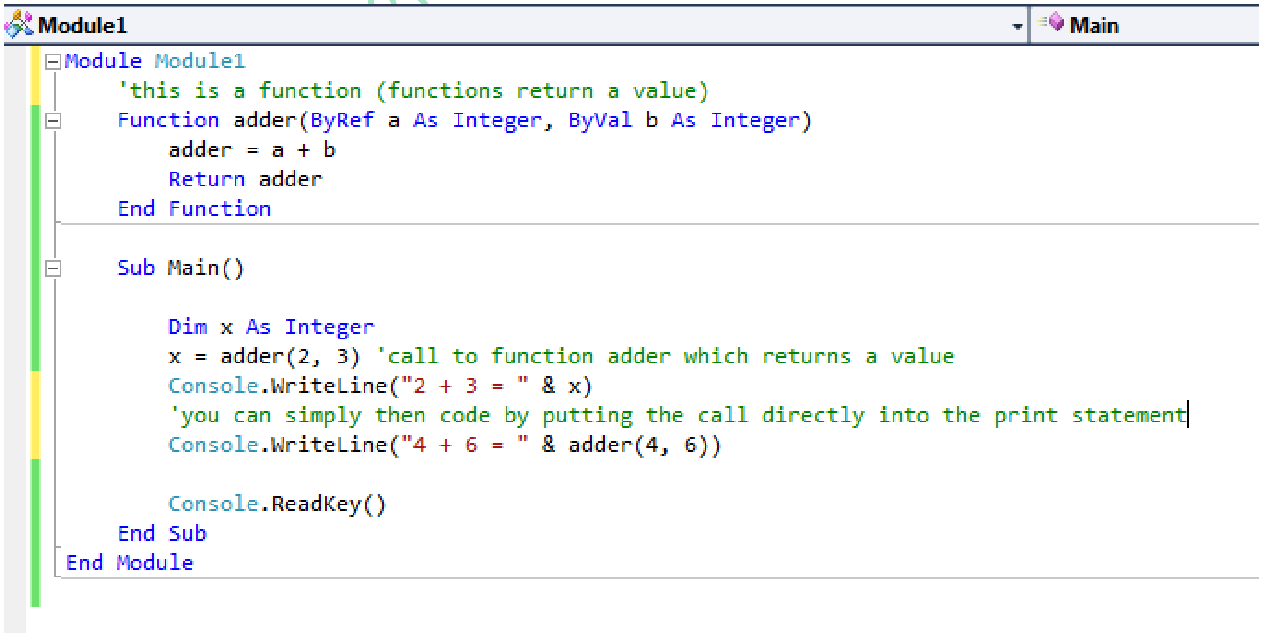
Functions

Functions are similar to Procedure, except that they always return a value..

```
FUNCTION square(num : INTEGER) : RETURNS INTEGER
        square = num * num
     RETURN square
END FUNCTION

FUNCTION sum (a :INTEGER, b : INTEGER) RETURNS INTEGER
        sum = a + b
     RETURN sum
END FUNCTION

BEGIN
     DECLARE number, value1, value2 : INTEGER
     PRINT ("Please Input a number for its square")
     INPUT number
     PRINT ("Square of number is: ")
     PRINT (CALL square(number))
     PRINT ("Please Input a num1 and num2 for sum ")
     INPUT value1, value2
     PRINT ("Sum is" (CALL sum(value1, value2))
END
```

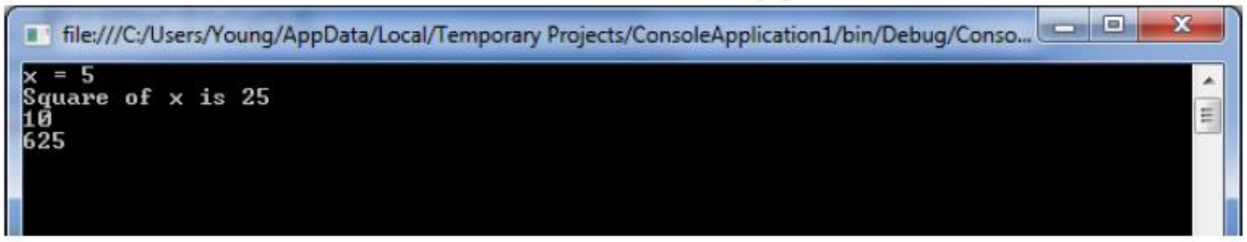
CS (2210) with Sir Majid Tahir at www.majidtahir.com

Example Program in VB - Functions



```
Module Module1
    'this is a function (functions return a value)
    Function adder(ByRef a As Integer, ByVal b As Integer)
        adder = a + b
        Return adder
    End Function

    Sub Main()

        Dim x As Integer
        x = adder(2, 3) 'call to function adder which returns a value
        Console.WriteLine("2 + 3 = " & x)
        'you can simply then code by putting the call directly into the print statement
        Console.WriteLine("4 + 6 = " & adder(4, 6))

        Console.ReadKey()
    End Sub
End Module
```

```
Module Module1
    Function square(ByVal x As Integer) As Integer
        square = x * x
    End Function

    Function sum(ByRef a As Integer, ByRef b As Integer) As Integer
        sum = a + b
    End Function

    Sub Main()
        Dim number As Double = 5
        Console.WriteLine("x = " & number)
        Console.WriteLine("Square of x is " & square(number))
        Console.WriteLine(sum(3, 7))
        Console.WriteLine(square(sum(16, 9)))
        Console.ReadLine()
    End Sub
End Module
```


file:///C:/Users/Young/AppData/Local/Temporary Projects/ConsoleApplication1/bin/Debug/Conso...

```
x = 5
Square of x is 25
10
625
```

Example

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

```
Function FindMax(ByVal num1 As Integer, ByVal num2 As Integer) As Integer ' local
variable declaration
    Dim result As Integer
    If (num1 > num2) Then
        result = num1
    Else
        result = num2
    End If
    FindMax = 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

Computer Sc. (9618) with M... + www.majidtahir.com

Function & Procedure in one Program (Pseudocode).

```
PROCEDURE grades(num As Integer)
    Case OF num
           >= 90 : OUTPUT ("A*")
           >= 80 : OUTPUT ("A")
           >= 70 : OUTPUT ("B")
           >= 60 : OUTPUT ("C")
        OTHERWISE OUTPUT ("Need improvement")
    End CASE
End PROCEDURE

PROCEDURE pass (number As Integer)
    IF number >= 50 THEN
        OUTPUT ("Pass")
    ELSE
        OUTPUT ("Fail")
    END IF
END PROCEDUE

FUNCTION regular (n: INTEGER) RETURNS : STRING
    IF n >= 60 THEN
        regular = ("Regular admission from school")
    ELSE
        regular = ("Appears as a Private candidate")
    END IF
    RETURN regular
END FUNCTION

BEGIN
    DECLARE marks : Integer
    OUTPUT ("Enter your marks")
    INPUT marks

    WHILE marks > 100 OR marks < 0
        OUTPUT ("Error, Re-Enter marks between 0 to 100")
        INPUT marks
    END WHILE

    CALL grades(marks)
    CALL pass(marks)
    OUTPUT (CALL regular(marks))
END
```

Computer Sc. (9618) with Majid Tahir at www.majidtahir.com
CS (2210) with Sir Majid at www.majidtahir.com

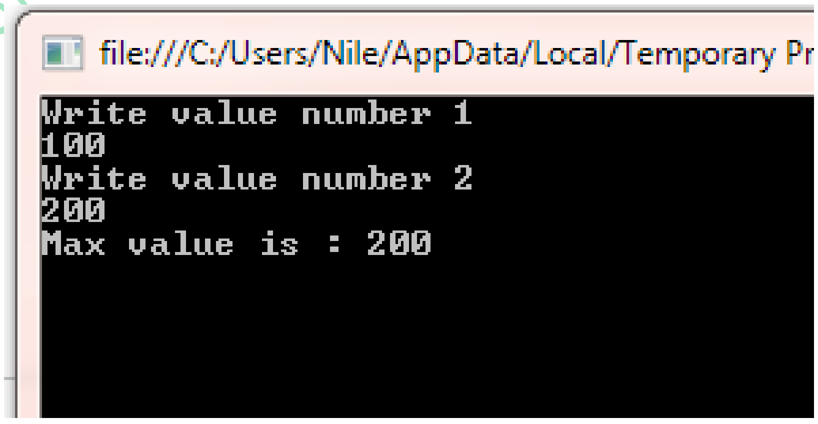
The following example demonstrates *FindMax* function in VB (Console mode):

```
Module module1
    Function FindMax(ByVal num1 As Integer, ByVal num2 As Integer) As Integer
        ' local variable declaration */
        Dim result As Integer
        If (num1 > num2) Then
            result = num1
        Else
            result = num2
        End If
        FindMax = result
    End Function
    Sub Main()
        Dim a, b, res As Integer
        Console.WriteLine("Write value number 1")
        a = Console.ReadLine()
        Console.WriteLine("Write value number 2")
        b = Console.ReadLine()

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

CS (2210) with Sir Majid at www.majidtahir.com

Computer Sc. (9618) with Majid Tahir at www.majidtahir.com

When the above code is compiled and executed, it takes value 1 & value 2 as input and produces the maximum value for example:



```
Write value number 1
100
Write value number 2
200
Max value is : 200
```

References:
Visual Basics Console Cook Book by
VB.NET Console Book by ***Dough Semple***
https://www.tutorialspoint.com/vb.net/vb.net_functions.htm
https://docs.microsoft.com/en-us/dotnet/visual-basic/language-reference/statements/function-statement