Visual Basic .NET 2010 Fundamentals

First Edition

E. A. Abdelhalem

|  |  |  |
| --- | --- | --- |
| Date: 19/02/2017 02:00 PM | Day: Sunday | Section Num.: (01)  |

1. **Creating Your First Visual Basic Program**

**HELLO WORLD**

1. Installing Visual basic .NET 2010 Express Edition.
2. Fell for the workflow.
3. How to create new project.
4. Where to type the code.
5. How to test the project.
6. Run and Stop debugging icons
7. Comment single quote
8. Member accessor operator
9. Indentation and remove spaces
10. Only one statement in the same line
11. Module, Subroutine
12. objects
13. show and hide solution explorer
14. project folder and debug
15. show and hide extinction & solution extension
* Create New Project





Module Module1

 Sub Main()

 Console.WriteLine("Hello World")

 Console.ReadLine()

 End Sub

End Module

* **Exercise:**

**Determine the Errors in the following codes**

**(A):**

Module Module1

 Sub Main()

 End Sub

 Console.WriteLine("Hello World")

 Console.ReadLine()

End Module

**(B):**

Module Module1

 Sub Main()

 Console. WriteLine(Hello World)

 Console.ReadLine ()

 End Sub

End Module

**(c):**

Module Module1

 Sub Main()

 Console.WriteLine(Hello World)

 Console.ReadLine ()

End Module1

**(D):**

Module Module1

 Sub Main

 Console. WriteLine(Hello World)

 Console.ReadLine ()

End Module

**(E) Complete the following Sentences**

1. **Object consists of …............................... and …………………………**
2. **The member function must be ended with ……………………………**
3. **Module (larger - smaller) than subroutine.**
4. **String Must be between …………………………….**
5. **Indention is…………………………………….**
6. **Comment must start with …………………………….**
7. **Dot "." in console.writeline("Emadeldeen") called ………………………..**
8. **When run visual basic program the first run subroutine called …………….**
9. **Could you edit the code during debugging ? (yes, No)**

**(F) write and run visual basic code that write on the console window your impression on the section then send the folder of project on the following email emade09+vb@gmail.com**

|  |  |  |
| --- | --- | --- |
| Date:  | Day: Wedensday  | Section Num.: (02)  |

**Quick Overview of the VB Express Edition IDE**

1. **File->new project (CTRL + N)**
2. **New project (CTRL + N)**
3. **File –>open project (CTRL + O)**
4. **Open project (CTRL + O)**
5. **File->recent projects and solutions**
6. **Recent projects**
7. **Show and hide deferent windows**
8. **Break points**

**Step Into**

Executes code one statement at a time.

When not in design mode, **Step Into** enters [break mode](https://msdn.microsoft.com/en-us/library/office/gg264568.aspx) at the current line of execution. If the statement is a call to a procedure, the next statement displayed is the first statement in the procedure.

At [design time](https://msdn.microsoft.com/en-us/library/office/gg264568.aspx), this menu item begins execution and enters break mode before the first line of code is executed.

If there is no current execution point, the **Step Into** command may appear to do nothing until you do something that triggers code, for example click on a document.

Toolbar button: . Keyboard shortcut: F8.

**Step Over**

Similar to **Step Into**. The difference in use occurs when the current statement contains a call to a procedure.

**Step Over** executes the procedure as a unit, and then steps to the next statement in the current procedure. Therefore, the next statement displayed is the next statement in the current procedure regardless of whether the current statement is a call to another procedure. Available in break mode only.

Toolbar button: . Keyboard shortcut: SHIFT+F8.

**Step Out**

Executes the remaining lines of a function in which the current execution point lies. The next statement displayed is the statement following the procedure call. All of the code is executed between the current and the final execution points. Available in break mode only.

Toolbar button: . Keyboard shortcut: CTRL+SHIFT+F8.

1. **Debug->window->local**
2. **Throw code line by line and watch local**
3. **Intellisense menu "ctrl + spacebar" , press frist letter for next word**
4. **Show subroutine details through over courser**
5. **Do not go to next word line if Intellisense not working**
6. **Show and hide code blocks**
7. **Asterisk on the name of file menu indicate to that file has not been saved yet**
8. **Yellow line indicates to line which have not saved yet.**
* **Exercise:**
1. **Show how to use Brake point in visual basic**
2. **What is the purpose of local window and how to show it**
3. **What is the deference between step into, step over, step out**
4. **What is the Intellisense menu? How to show it ? how to use an item from it?**
5. **How to show details and purpose of subroutine?**
6. **How to show and hide code blocks statements**
7. **What are the following indicate to (asterisk, yellow line, green line)**
8. **Send an email to** emade09+vb@gmail.com **about you impression on this section.**

|  |  |  |
| --- | --- | --- |
| Date: | Day: Wedensday  | Section Num.: (03)  |

**Variables and Readline**

* **Variables**

# Visual Basic Naming Rules

Module Module1

 Sub Main()

 Dim x As Integer = 5

 Dim y As Double = 5.5

 Dim name As String = "Emadeldeen Abdallah"

 Console.WriteLine("your name is: {0}; x= {1}; y= {2} ", name, x, y)

 Console.ReadLine()

 End Sub

End Module







1. You must use a letter as the first character.
2. You can't use any spaces.
3. You can't use a space, period (**.**), exclamation mark (**!**), or the characters **@**, **&**, **$**, **#** in the name.
4. Name can't exceed 255 characters in length.
5. Generally, you shouldn't use any reserved words, that are the same as the [functions](https://msdn.microsoft.com/en-us/library/office/gg264568.aspx), [statements](https://msdn.microsoft.com/en-us/library/office/gg264568.aspx), and [methods](https://msdn.microsoft.com/en-us/library/office/gg264568.aspx) in Visual Basic
6. You can't reuse variable names in the same programe .
* **Examples**
1. Assuming there are 7.481 gallons in a cubic foot, write a program that asks the user to enter a number of gallons, and then displays the equivalent in cubic feet. Where the output is as the following.



Module Module1

 Sub Main()

 Dim foot As Double = 7.481

 Dim gallon As Double

 Console.WriteLine("Insert Number of gallons: ")

 gallon = Console.ReadLine()

 Console.WriteLine("{0} gallons = {1} Foot", gallon, gallon / foot)

 Console.ReadLine()

 End Sub

End Module

1. Write a program that generates the following table:



Module Module1

 Sub Main()

 Console.WriteLine("year" & vbTab & "value" & vbNewLine)

 Console.WriteLine("1990" & vbTab & "135")

 Console.WriteLine("1991" & vbTab & "7290")

 Console.WriteLine("1992" & vbTab & "11300")

 Console.WriteLine("1993" & vbTab & "16200")

 Console.ReadLine() 'Do not allow console close

 End Sub

End Module

1. Write program that read age as integer, percentage of success as double and name as string then the out as



Module Module1

 Sub Main()

 Dim x As Integer = 5

 Dim y As Double = 5.5

 Dim name As String = "Emadeldeen Abdallah"

 Console.WriteLine("your name is: {0}; x= {1}; y= {2} ", name, x, y)

 Console.ReadLine()

 End Sub

End Module

1. Write program that radius as double define pi () as a constant to calculate area and circumference of a circle where

Where the output is



Module Module1

 Sub Main()

 Const pi As Double = 22 / 7

 Dim radius, area, circumference As Double

 Console.WriteLine("Insert radius of the circle ")

 radius = Console.ReadLine()

 area = pi \* radius \* radius

 circumference = 2 \* pi \* radius

 Console.WriteLine("-----------------------")

 Console.WriteLine("area = {0}", area)

 Console.WriteLine("circumference = {0}", circumference)

 Console.ReadLine()

 End Sub

End Module

* **Exercise:**
1. **Complete the following**
2. To write tab in vb.net use …………...... but to write new line use …………………….
3. To concatenate two string use …………………….
4. To declare x as string write ……………….
5. To read x from user write …………………..
6. Write program that read a, b, c as double to calculate x (circumference) and area of triangle. Where

x = a + b + c

where the output is



Imports System.Math

Module Module1

 Sub Main()

 Const pi As Double = 22 / 7

 Dim a, b, c, x, area As Double

 Console.WriteLine("Insert a of the traingl")

 a = Console.ReadLine()

 Console.WriteLine("Insert b of the traingl")

 b = Console.ReadLine()

 Console.WriteLine("Insert c of the traingl")

 c = Console.ReadLine()

 x = a + b + c

 area = Sqrt(x \* (x - a) \* (x - b) \* (x - c))

 Console.WriteLine("-----------------------")

 Console.WriteLine("circumference = {0}", x)

 Console.WriteLine("area = {0}", area)

 Console.ReadLine()

 End Sub

End Module

1. Write program that read inchs and convert it to CM where 1cm = 2.54 inch
2. Write program that read x and print z where

Z = (2a - 4)7

an = a^n in visual basic ^ = shift + 6

1. Write program that print the following results



1. Write program to solve two linear equations of the first degree in two variables

Consider the two following equations

aX + bY – c = 0

dX + eY – f = 0

where

X = (c\*e-f\*b)/(a\*e-b\*d)

Y = (c\*d-f\*a)/(d\*b-a\*e)

|  |  |  |
| --- | --- | --- |
| Date | Day: Wedensday  | Section Num.: (04)  |

**Branching with the If..Then..Else Decision Statement**

Module Module1

 Sub Main()

 Dim userValue As String

 Console.WriteLine("Type any thing then press enter")

 userValue = Console.ReadLine()

 Console.WriteLine("you typed: {0}", userValue)

 Console.ReadLine()

 End Sub

End Module

* If statment

If (condition) Then

 'statments

End If

**Example (01):**

Module Module1

 Sub Main()

 Dim userValue As String

 Console.WriteLine("Type any thing then press enter")

 userValue = Console.ReadLine()

 If (userValue = "mypassword") Then

 Console.WriteLine("you typed: {0}", userValue)

 End If

 Console.ReadLine()

 End Sub

End Module

**Example (02):** write program that define constant pass as string and variable userValue as string then ask user to insert password, if pass = userValue write "you loged in"



Module Module1

 Sub Main()

 Const pass As String = "emade09"

 Dim userValue As String

 Console.WriteLine("Insert password thin press enter : ")

 userValue = Console.ReadLine()

 If (pass = userValue) Then

 Console.WriteLine("you loged in")

 End If

 Console.ReadLine()

 End Sub

End Module

* If ….. else…… statement

 If (condision) Then

 'write statments her

 Else

 'write else statments her

 End If

**Example (03):** write program that define constant pass as string and variable userValue as string then ask user to insert password, if pass = userValue write "you loged in" else write "wrong password! Try again" where the output is





Imports System.Math

Module Module1

 Sub Main()

 Const pass As String = "emade09"

 Dim userValue As String

 Console.WriteLine("Insert password thin press enter : ")

 userValue = Console.ReadLine()

 If (pass = userValue) Then

 Console.WriteLine("you loged in")

 Else

 Console.WriteLine("wrong password! Try again")

 End If

 Console.ReadLine()

 End Sub

End Module

* If ….. elseif……else statement

If (condision1) Then

 'write statments1 her

Else If (condision2) Then

 'write elseif statments2 her

Else If (condision3) Then

 'write elseif statments3 her

 Else

 'write else statments her

 End If

**Example (04):** write program that ask user to write his name then program print for him his salary but if name is not found print sorry "Wrong name! Try agan" where

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Emad | Nada | Ahmed | Samer | Nour |
| 1500 | 1250 | 950 | 1750 | 1800 |

 The output as like that



Module Module1

 Sub Main()

 Dim userName As String

 Console.WriteLine("Insert user name : ")

 userName = Console.ReadLine()

 If (userName = "Emad") Then

 Console.WriteLine("you salary is 1500")

 ElseIf (userName = "Nada") Then

 Console.WriteLine("you salary is 1250")

 ElseIf (userName = "Ahmed") Then

 Console.WriteLine("you salary is 950")

 ElseIf (userName = "Samer") Then

 Console.WriteLine("you salary is 1750")

 ElseIf (userName = "Nour") Then

 Console.WriteLine("you salary is 1800")

 Else

 Console.WriteLine("Wrong name! Try agan")

 End If

 Console.ReadLine()

 End Sub

End Module

* Logical operators

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A | B | A And B | A Or B | Not (A) | Not (B) |
| True | True | True | True | False | False |
| True | False | False | True | False | True |
| False | True | False | True | False | False |
| False | False | False | False | False | True |

* Comparison Operators



* Mod Operator

number1 Mod number2

5 mod 3 = 2

5 mod 10 = 5

5 mod 4 = 1

5 mod 2 = 1

5 mod 5 = 0

Example (05): write program that as user to insert integer number then print "Even number" if the number is even else print "Odd number" where the output like this:



 **Note use** If ((x Mod 2) = 0)

Example (06): write program that read three numbers a, b, c as integer from user then print the maximum value where the output like this



Module Module1

 Sub Main()

 Dim a, b, c As Integer

 Console.WriteLine("a = ")

 a = Console.ReadLine()

 Console.WriteLine("b = ")

 b = Console.ReadLine()

 Console.WriteLine("c = ")

 c = Console.ReadLine()

 If ((a > b) And (a > c)) Then

 Console.WriteLine("a= {0} is maximam value", a)

 ElseIf ((b > a) And (b > c)) Then

 Console.WriteLine("b= {0} is maximam value", b)

 Else

 Console.WriteLine("c= {0} is maximam value", c)

 End If

 Console.ReadLine()

 End Sub

End Module

* **Exercise:**
1. write program to find maximum value of two variables.
2. Write program to find The roots of a quadratic equation in one variable like this : aX2 + bX + c = 0

|  |  |  |
| --- | --- | --- |
| Date | Day: Wedensday  | Section Num.: (05)  |

**Branching with Select Case Statement**

 Select Case expression

 Case Is condition1

 Code bloke1 of condition1 is true

 Case Is condition2

 Code bloke2 of condition2 is true

 Case Is condition3

 Code bloke3 of condition3 is true

 Case Else

 Statements at default case, when all cases are invalide

 End Select

****

Example (01): Write program that read student's mark as integer then print the equivalent grade depends on the following table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0≤Mark<60 | 60≤Mark<65 | 65≤Mark<75 | 75≤Mark<85 | 85≤Mark<100 |
| Fail | OK | Good | Very Good | Excellent |

Module Module1

 Sub Main()

 Dim grade As Integer

 grade = Console.ReadLine()

 Select Case grade

 Case Is < 60

 Console.WriteLine("Fail")

 Case Is < 65

 Console.WriteLine("OK")

 Case Is < 75

 Console.WriteLine("Good")

 Case Is < 85

 Console.WriteLine("Very Good")

 Case Is < 100

 Console.WriteLine("Excellent")

 Case Else

 Console.WriteLine("Invalid Mark! Try again!")

 End Select

 Console.ReadLine()

 End Sub

End Module

 

Example (02): Rewrite the following program using Select Case Statement:

Module Module1

 Sub Main()

 Dim userName As String

 Console.WriteLine("Insert user name : ")

 userName = Console.ReadLine()

 If (userName = "Emad") Then

 Console.WriteLine("you salary is 1500")

 ElseIf (userName = "Nada") Then

 Console.WriteLine("you salary is 1250")

 ElseIf (userName = "Ahmed") Then

 Console.WriteLine("you salary is 950")

 ElseIf (userName = "Samer") Then

 Console.WriteLine("you salary is 1750")

 ElseIf (userName = "Nour") Then

 Console.WriteLine("you salary is 1800")

 Else

 Console.WriteLine("Wrong name! Try agan")

 End If

 Console.ReadLine()

 End Sub

End Module

**After Converting**

Module Module1

 Sub Main()

 Dim userName As String

 Console.WriteLine("Insert user name : ")

 userName = Console.ReadLine()

 Select Case userName

 Case Is = "Emad"

 Console.WriteLine("you salary is 1500")

 Case Is = "Nada"

 Console.WriteLine("you salary is 1250")

 Case Is = "Ahmed"

 Console.WriteLine("you salary is 950")

 Case Is = "Samer"

 Console.WriteLine("you salary is 1750")

 Case Is = "Nour"

 Console.WriteLine("you salary is 1800")

 Case Else

 Console.WriteLine("Wrong name! Try agan")

 End Select

 Console.ReadLine()

 End Sub

End Module

* Exercise (01): Write the output of the following code:

Module Module1

 Sub Main()

 Dim a As Integer = 100

 Dim b As Integer = 200

 Select Case a

 Case 100

 Console.WriteLine("This is part of outer case ")

 Select Case b

 Case 200

 Console.WriteLine("This is part of inner case ")

 End Select

 End Select

 Console.WriteLine("Exact value of a is : {0}", a)

 Console.WriteLine("Exact value of b is : {0}", b)

 Console.ReadLine()

 End Sub

End Module

* Exercise (02): Write program that read a word English vocabulary from user as string and print its definition, at case of word not found print "The vocabulary is not found in the database".

Note: solve using if statement then try to solve using select case statement

|  |  |
| --- | --- |
| **word vocabulary** | **Definition** |
| Pillow | Is soft or hard thing that you put your head on it. |
| Pillowcase  | Is a cover for the pillow |
| Dresser  | Is a piece of furniture that you put your clothes into  |

* Exercise (03): Write the output of the following code Then rewrite it using select case statement:

Module Module1

 Sub Main()

 Dim a As Integer = 100

 Dim b As Integer = 200

 If (a = 100) Then

 If (b = 200) Then

 Console.WriteLine("Value of a is 100 and b is 200")

 End If

 End If

 Console.WriteLine("Exact value of a is : {0}", a)

 Console.WriteLine("Exact value of b is : {0}", b)

 Console.ReadLine()

 End Sub

End Module

|  |  |  |
| --- | --- | --- |
| Date | Day:  | Section Num.: (06)  |

**Loops (For Loop)**



For a = **startValue** To **endValue** Step **stepValue**

 Console.WriteLine("value of a: {0}", a)

Next

* Example (01): **Write program that prints numbers from 1 to 100:**

Module Module1

 Sub Main()

 Dim i As Integer

 For i = 1 To 100

 Console.WriteLine(i)

 Next

 Console.ReadLine()

 End Sub

End Module

* Example (02): **Write program that prints odd numbers from 1 to 100:**

Module Module1

 Sub Main()

 Dim i As Integer

 For i = 1 To 100 Step 2

 Console.WriteLine(i)

 Next

 Console.ReadLine()

 End Sub

End Module

* Example (03): **Write program that prints even numbers from 1 to 100:**

Module Module1

 Sub Main()

 Dim i As Integer

 For i = 0 To 100 Step 2

 Console.WriteLine(i)

 Next

 Console.ReadLine()

 End Sub

End Module

* Example (04): **Write program that prints numbers divided by 7 from 1 to 100:**

Module Module1

 Sub Main()

 Dim i As Integer

 For i = 1 To 100

 If ((i Mod 7) = 0) Then

 Console.WriteLine(i)

 End If

 Next

 Console.ReadLine()

 End Sub

End Module

* Example (05): Nested ……. For: **Write program that prints the following output using nested for**



Module Module1

 Sub Main()

 Dim i, j As Integer

 For i = 1 To 3

 For j = 1 To 3

 Console.Write("{0}" & vbTab, i \* j)

 Next

 Console.WriteLine()

 Next

 Console.ReadLine()

 End Sub

End Module

* Example (06): **Write program that find the summation of numbers from 1 to 50**

Module Module1

 Sub Main()

 Dim i, sum As Integer

 sum = 0

 For i = 1 To 50

 sum = sum + i

 Next

 Console.WriteLine("summision of 1 to 50 is = {0}", sum)

 Console.ReadLine()

 End Sub

End Module

* Example (07): **Write program that read n as integer and find the product of n.**

Module Module1

 Sub Main()

 Dim i, pro, n As Integer

 Console.Write("write a number to find the product:")

 n = Console.ReadLine()

 pro = 1

 For i = 1 To n

 pro = pro \* i

 Next

 Console.WriteLine("product of {0} is = {1}", n, pro)

 Console.ReadLine()

 End Sub

End Module

* Example (08): **Write program that read n as integer and find:**

Module Module1

 Sub Main()

 Dim i, sum, n As Integer

 Console.Write("Insert n to find the sum of (2i+1)^7 where i=1 to n :")

 n = Console.ReadLine()

 sum = 1

 For i = 1 To n

 sum += (2 \* i + 1) ^ 7

 Next

 Console.WriteLine("The sum of (2i+1)^7 where i=1 to {0} ={1}", n, sum)

 Console.ReadLine()

 End Sub

End Module

* Example (09): **Write program that read "n" as integer then prints Fibonacci series of "n" as: 0 1 1 2 3 5 8 13 21 34 ……………..**

 Sub Main()

 Dim i, n, f, f1, f2 As Integer

 Console.Write("Insert integer number to print fibonacci:")

 n = Console.ReadLine()

 f1 = 0

 f2 = 1

 For i = 1 To n

 Console.Write("{0}" & vbTab, f1)

 f = f2

 f2 = f1 + f2

 f1 = f

 Next

 Console.ReadLine()

 End Sub

* Exercises
1. **Write program that read "n" to find summation of:**
2. **Write program that read "n" to find summation of:**
3. **Write program that read n of numbers then print the multiplication table of n until n\*12 as the following.**



1. **Write program that read n of numbers then prints the mean of numbers.**
2. **Write program that read n of numbers then prints the maximum value.**
3. **Write program that read n then print "n is prime" if n is prime number else print "n is not prime" if n is not prime number.**
4. **Write program that prints prime numbers from 1 to 100.**
5. **Write program that read n of numbers then count and print:**

**Number of positive numbers,**

**The summation of positive numbers,**

**Number of negative numbers,**

**The summation of negative numbers,**

**Number of numbers that equal zero.**

**as the following:**



Module Module1

 Sub Main()

 Dim n As Integer ' number of numbers you want to insert

 Dim x As Double ' the number you will insert

 Dim np As Integer = 0 ' Number of Positive (np)

 Dim sp As Double = 0 ' Summasion of Positive (sp)

 Dim n0 As Integer = 0 ' Number of zero (n0)

 Dim nn As Integer = 0 ' Number of negative (nn)

 Dim sn As Double = 0 ' Summasion of negative (sn)

 Console.Write("Insert n of numbers to insert it:")

 n = Console.ReadLine()

 Console.WriteLine("--------------------------------------")

 For i = 1 To n

 Console.Write("Number{0} = ", i)

 x = Console.ReadLine()

 If (x > 0) Then

 np = np + 1

 sp = sp + x

 ElseIf (x < 0) Then

 nn = nn + 1

 sn = sn + x

 Else

 n0 = n0 + 1

 End If

 Next

 Console.WriteLine("--------------------------------------")

 Console.WriteLine("Number of Positive = {0}", np)

 Console.WriteLine("Summasion of Positive = {0}", sp)

 Console.WriteLine("Number of zeros = {0}", n0)

 Console.WriteLine("Number of negative = {0}", nn)

 Console.WriteLine("Summasion of negative = {0}", sn)

 Console.ReadLine()

 End Sub

End Module

|  |  |  |
| --- | --- | --- |
| Date | Day:  | Section Num.: (07)  |

**Arrays**



* Define an array in vb.net

Dim intData(30) ' an array of 31 elements

Dim strData(20) As String ' an array of 21 strings

Dim twoDarray(10, 20) As Integer 'a two dimensional array of integers

Dim ranges(10, 100) 'a two dimensional array

Dim intData() As Integer = {12, 16, 20, 24, 28, 32}

Dim names() As String = {"Karthik", "Sandhya", "Shivangi", "Ashwitha", "Somnath"}

* Example (01): **Write program that read 7 numbers then prints the mean of numbers.**

Module Module1

 Sub Main()

 Dim i As Integer 'the counter of for loop

 Dim sum As Double = 0 'to store the summasion of numbers

 Dim nums(7) As Double 'array of 7 indexs to numbers

 For i = 1 To 7

 Console.Write("Insert Num {0}: ", i)

 nums(i) = Console.ReadLine()

 Next

 Console.WriteLine("------------------------------------")

 For i = 1 To 7

 sum = sum + nums(i)

 Next

 Console.WriteLine("The summasion is ={0}", sum)

 Console.ReadLine()

 End Sub

End Module

* **Dynamic Arrays**

Dim marks(1) As Integer

marks(0) = 85

marks(1) = 75

**ReDim Preserve marks(4)**

marks(2) = 80

marks(3) = 76

* Example (02): **Write program that read n of numbers then prints the maximum value.**

Module Module1

 Sub Main()

 Dim i As Integer 'the counter

 Dim n As Integer 'numbers of Inserted Numbers

 Dim max As Double 'to store the maximum value

 Dim nums() As Double 'array of 0 indexs to numbers

 Console.Write("Insert Number on Inserted Numbers:")

 n = Console.ReadLine()

 ReDim Preserve nums(n)

 For i = 1 To n

 Console.Write("Number {0} = ", i)

 nums(i) = Console.ReadLine()

 Next

 max = nums(1)

 For i = 1 To n

 If (nums(i) > max) Then

 max = nums(i)

 End If

 Next

 Console.WriteLine("The maximum number is ={0}", max)

 Console.ReadLine()

 End Sub

End Module

* Example (03): **Write program that read matrix of n\*n of double numbers then prints the summation of diameter of the matrix.**



Module Module1

 Sub Main()

 Dim row, col As Integer 'counter of rows and columns

 Dim len As Integer 'length of the array

 Dim sum As Double = 0 'to store the summasion of numbers

 Dim nums(,) As Double 'array of 0\*0

 Console.Write("Insert length of the array:")

 len = Console.ReadLine()

 ReDim Preserve nums(len, len)

 Console.WriteLine("Insert elements of the array({0}, {1}):", len, len)

 For row = 1 To len

 For col = 1 To len

 Console.Write("Array({0}, {1}) = ", row, col)

 nums(row, col) = Console.ReadLine()

 Next

 Next

 For row = 1 To len

 For col = 1 To len

 If (row = col) Then

 sum = sum + nums(row, col)

 End If

 Next

 Next

 Console.WriteLine("The summasion of diameter is ={0}", sum)

 Console.ReadLine()

 End Sub

End Module

* Example (04): **Write program that read matrix of n\*n of double numbers then prints the summation of upper triangle of the matrix.**



Module Module1

 Sub Main()

 Dim row, col As Integer 'counter of rows and columns

 Dim len As Integer 'length of the array

 Dim sum As Double = 0 'to store the summasion of numbers

 Dim nums(,) As Double 'array of 0\*0

 Console.Write("Insert length of the array:")

 len = Console.ReadLine()

 ReDim Preserve nums(len, len)

 Console.WriteLine("Insert elements of the array({0}, {1}):", len, len)

 For row = 1 To len

 For col = 1 To len

 Console.Write("Array({0}, {1}) = ", row, col)

 nums(row, col) = Console.ReadLine()

 Next

 Next

 For row = 1 To len

 For col = 1 To len

 If (row > col) Then

 sum = sum + nums(row, col)

 End If

 Next

 Next

 Console.WriteLine("The summasion of diameter is ={0}", sum)

 Console.ReadLine()

 End Sub

End Module

|  |  |  |
| --- | --- | --- |
| Date | Day:  | Section Num.: (08)  |

**Functions & subroutine**

Function functionName() As Type

 'Statements

 functionName = value of Type

End Function

* Example (01): **Write function** charRepeat **that repeat '\*' five times.**

Module Module1

 Sub Main()

 Dim str As String = charRepeat()

 Console.WriteLine(str)

 Console.ReadLine()

 End Sub

 Function charRepeat() As String

 charRepeat = ""

 For i As Integer = 1 To 5

 charRepeat = charRepeat & "\*"

 Next

 End Function

End Module

Function functionName(ByVal var As Type) As Type

 'Statements

 functionName = value of Type

End Function

* Example (02): **Write function** charRepeat **that take n as integer parameter then return '\*' repeated n of times.**

Module Module1

 Sub Main()

 Dim str As String = charRepeat(6)

 Console.WriteLine(str)

 Console.ReadLine()

 End Sub

 Function charRepeat(ByVal n As Integer) As String

 charRepeat = ""

 For i As Integer = 1 To n

 charRepeat = charRepeat & "\*"

 Next

 End Function

End Module

Function functionName(ByVal var1 As Type**,** ByVal var2 As Type) As Type

 'Statements

 functionName = value of Type

End Function

* Example (03): **Write function** charRepeat **that take n as integer and c as char parameter then return c repeated n of times.**

Module Module1

 Sub Main()

 Dim str As String = charRepeat("#", 10)

 Console.WriteLine(str)

 Console.ReadLine()

 End Sub

 Function charRepeat(ByVal c As Char, ByVal n As Integer) As String

 charRepeat = ""

 For i As Integer = 1 To n

 charRepeat = charRepeat & c

 Next

 End Function

End Module

* Example (04): **Write function factorial that take n as integer then return the factorial of n.**

Module Module1

 Sub Main()

 Dim str As integer = factorial(5)

 Console.WriteLine(str)

 Console.ReadLine()

 End Sub

 Function factorial(ByVal n As Integer) As Integer

 factorial = 1

 For i As Integer = 2 To n

 factorial = factorial \* i

 Next

 End Function

End Module

* Example (05): **Write function power that take x and n as integers then return xn.**

Module Module1

 Sub Main()

 Dim str As Integer = power(2, 5)

 Console.WriteLine(str)

 Console.ReadLine()

 End Sub

 Function power(ByVal x As Integer, ByVal n As Integer) As Integer

 power = 1

 For i As Integer = 1 To n

 power = power \* x

 Next

 End Function

End Module

* Exercises
1. **Write function that take "n" as parameter to return the summation of:**
2. **Write function that take "n" as parameter to return the summation of:**
3. **Write subroutine that take "n" as parameter then print the multiplication table of n until n\*12 as the following.**



1. **Write function that take "n" as integer parameter then return "n is prime" if n is prime number else return "n is not prime" if n is not prime number.**
2. **Write subroutine that take x and y as integers parameter then prints prime numbers from x to y.**