1. **Nested Loops**

VB.Net allows using one loop inside another loop. Following section shows few examples to illustrate the concept.

The syntax for a **nested For loop** statement in VB.Net is as follows −

For counter1 [ As datatype1 ] = start1 To end1 [ Step step1 ]

For counter2 [ As datatype2 ] = start2 To end2 [ Step step2 ]

Statements

Next [ counter2 ]

Next [ counter 1]

The syntax for a **nested While loop** statement in VB.Net is as follows −

While condition1

While condition2

Statements

End While

End While

The syntax for a **nested Do...While loop** statement in VB.Net is as follows −

Do { While | Until } condition1

Do { While | Until } condition2

Statements

Loop

Loop

A final note on loop nesting is that you can put any type of loop inside of any other type of loop. For example, **For-Loop** can be inside a **While Loop** or vice versa.

**Note:** Tools that used with selection statements are ListBox and ComboBox.

**H.W:** By using nested loop concepts design and write the proper codes to display a structure of stars “\*” in triangle format using label control.

**\***

**\*\***

**\*\*\***

**\*\*\*\***

**\*\*\*\*\***

**\*\*\*\*\*\***

## Some of Important Functions & Properties & Methods

## String Functions

## 1. Len: Return length of the string. Example: Len(“Ira q”) = 5

## 2. Mid: Return substring containing a specified number of characters.

## Mid(string, position, length)

## Example: mid(“ bai da“, 3, 2) = ai رمز الفراغ يحتسب من ضمن طول الخيط الرمز

## 3. Ltrim: Return a copy of string without any spaces from left side.

## Example: Left(“ bai da “) = “baida “

## Rtrim: Return a copy of string without any spaces from right side.

## Example: Left(“ bai da “) = “ baida”

## Trim: Return a copy of string without any spaces from both sides left & right

## Example: Left(“ bai da “) = “baida”

## Ucase & L case: Ucase convert all characters of string to the capital letters, while the Lcase convert all characters of string to the small letters.

## Ucase(“baida”) = BAIDA

## Space: used to insert number of spaces as required.

## Space(5)

## Math Functions

## Fix: Returns the integer part of the number.

## Example : Fix (7.1234) = 7

## Rnd: Returns a random value between 0 & 1. Random numbers often need to be converted into integers in programming. For example if we wish to obtain a random output of 6 integers ranging from 1 to 6, we need to convert the random numbers to integers using the following formula Int(Rnd\*6)+1.

## Date & Time

## Now: Returns date & time according to your system

## Today: Returns date according to your system

## TimeOfDay: Returns time according to your system

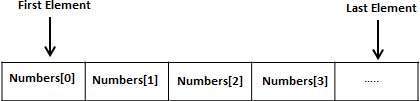
## Month: Returns the month of any date. Example: Month (“1996/6/25”) = 6

## Year: Returns the year of any date. Example: Year (now) = 2020

## Day: Returns the day of any date. Example: Day (today) = 25

## The Arrays

* **One Dimension Arrays**

An array stores a fixed-size sequential collection of elements of the same type. It is used to store a collection of data. All arrays consist of contiguous memory locations. The lowest address corresponds to the first element and the highest address to the last element.

## Declaring Arrays in VB.Net

To declare an array in VB.Net, you can use the **Dim** statement. For example,

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

You can also initialize the array elements while declaring the array. For example,

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

Dim names( ) As String = {"Karth", "Sandy", "Shivangi", "Shwitha", "Somnath"}

Dim Names(2) as String

Names(0) = “Ali”

Names(1) = “Ahmed”

Names(2) = “Sara”

The elements in an array can be stored and accessed by using the index of the array. The following program demonstrates this –

Private Sub Button4\_Click( )

Dim n(10) As Integer ' n is an array of 11 integers '

Dim i, j As Integer

' initialize elements of array n '

For i = 0 To 10

n(i) = i + 100 ' set element at location i to i + 100

Next i

' Output each array element's value '

For j = 0 To 10

Label1.Text = Label1.Text & "Element (" & j & ")" & Space(4) & n(j) &\_

vbNewLine

Next j

End Sub

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

Element(0) = 100

Element(1) = 101

Element(2) = 102

Element(3) = 103

Element(4) = 104

Element(5) = 105

Element(6) = 106

Element(7) = 107

Element(8) = 108

Element(9) = 109

Element(10) = 110

**Appendix 1**

* + - 1. Listbox1.Items.Add (“ALI”) Or (InputBox(“EnterValue ..?”) Or others {it will insert the items at the end of list}
      2. Listbox1.Items.Remove (“ALI”) {it will delete specified item}
      3. Listbox1.Items.RemoveAt (Listbox1.selectedindex) Or (2) or (Listbox1.items.count-i) Or others {it will delete item after select it
      4. Listbox1.Items.count. {retrieve number of items in the list}
      5. Listbox1.Items.Insert (i, ”any text”) {It will insert item at the position i}
      6. Listbox1.Items.contain (“ any text”) {it will return True if found specific text}
      7. Listbox1.sorted = True {it will arrange the list}