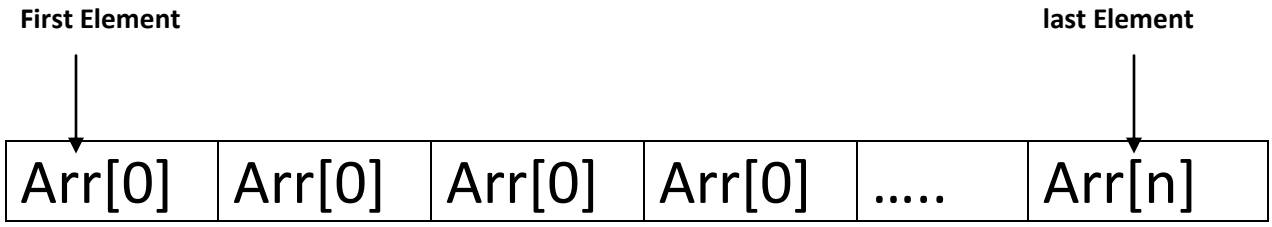
Array in VB.NET

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

Introduction :

An **array** is a linear data structure that is a collection of data elements of the same type stored on a **contiguous memory** location. Each data item is called an element of the array. It is a fixed size of sequentially arranged elements in computer memory with the first element being at **index 0** and the last element at index **n,** where **n+1** represents the total number of elements in the array.

The following is an illustrated representation of similar data type elements defined in the VB.NET array data structure.



Array can have either one dimension or multiple dimensions. An array declared with one dimension is termed as single dimensional array. Array declared with more than one dimension can be termed as multidimensional array.

SingleDimensional Array

Declaration of VB.NET Array

We can declare an array with the help of **DIM** statement. We need to specify the number of the elements followed by parentheses () in the VB.NET.

Syntax :

Dim <Array Name(upper bound)> as <data type>

In the above declaration, **Array_name** is the name of an array, and the **Data_Type** represents the type of element (Integer, char, String, Decimal) that will to store contiguous data elements in the VB.NET array.

***upper bound :** index value of the Last element. As index value of the first element in 0 therefore the size of the array is **[upper bound +1]**

Example :

dim arr(5) as integer

Here an array of name arr is declared of size **6**. Type of the array is **integer**

dim nm(10) as String

Here an array of name nm is declared of size **11**. Type of the array is **String**

Initialization of VB.NET Array

In VB.NET, we can initialize an array with **New** keyword at the time of declaration.

syntax :

```
Dim arr As Integer() = New Integer(5) {1, 2, 3, 4, 5, 6}
```

Program to create an array of size 5. Initialize it and display the array in reverse order.

```
Dim a As Integer() = New Integer(5) {1, 2, 3, 4, 5, 6}
 Dim i
 For i = 5 To 0 Step -1
     MsgBox(a(i))
 Next
```

Multidimensional Array

In VB.NET, a multidimensional array is useful for storing more than one dimension in a tabular form, such as rows and columns. The multidimensional array support two or three dimensional in VB.NET.

Declaration of VB.NET Array

Dim <Array Name(upper bound of column)(upper bound of row)> as <data type>

Example :

dim arr(2)(5) as integer

Here an array of name arr is declared of size **3X6 (Number of row = 3 and number of column = 6)**. Type of the array is **integer**

Initialization of double dimensional Array

In VB.NET, we can initialize an array with **New** keyword at the time of declaration.

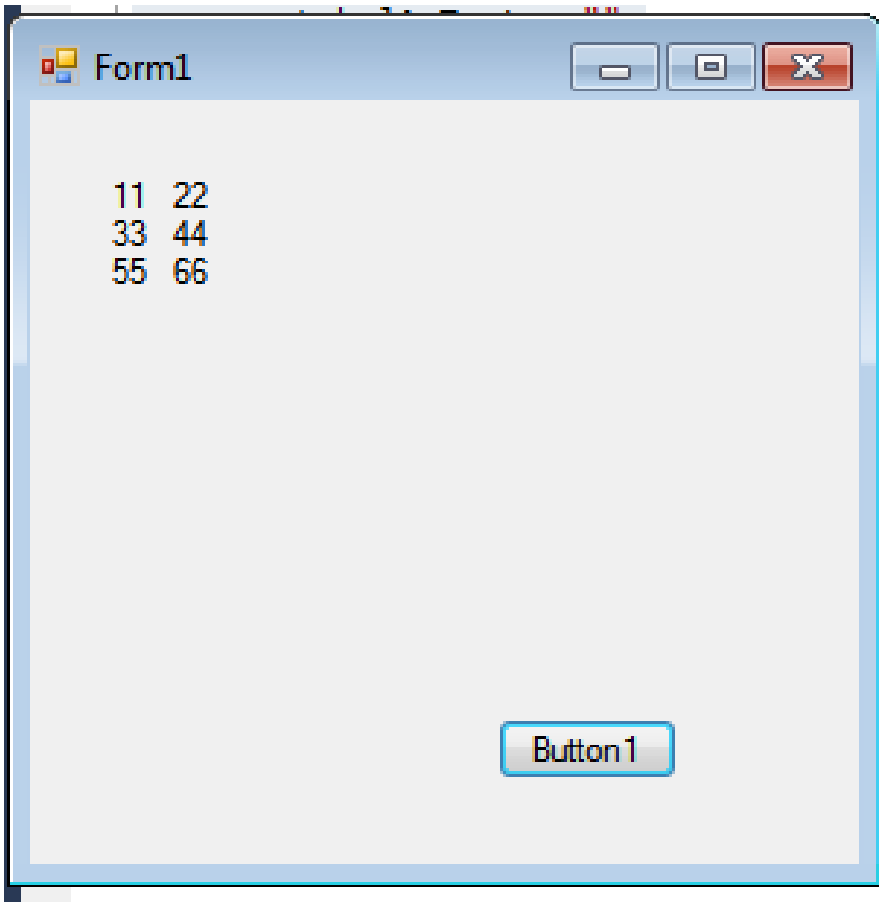
syntax :

```
Dim arr(,) As Integer = {{11, 22}, {33, 44}, {55, 66},
                         {77, 88}, {99, 11}}
```

Program to create an array of size 2x3. Initialize it and display the same.

```
  Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
        Label1.Text = ""
        Dim arr(,) As Integer = {{11, 22}, {33, 44}, {55, 66}}
        Dim i, j
        For i = 0 To 2
            For j = 0 To 1
                Label1.Text = Label1.Text & arr(i, j) & "   "
            Next j
            Label1.Text = Label1.Text & vbCrLf
        Next i
```

output



Fixed Size Array

In VB.NET, a fixed- size array is used to hold a fixed number of elements in memory. It means that we have defined the number of elements in the array declaration that will remain the same during the definition of the elements, and its size cannot be changed.

Dynamic Array

This is an array that can hold any number of elements. The array size can grow at any time. This means that you can add new elements to the array any time we want.

Redim Statement

For Dynamic array declaration we need Redim statement. First we need to declare an array with dim statement without specifying its size. Later on Using Redim we can specify it size like this

Syntax :

Redim [Preserve] array-name (subscripts)

example :

Dim a() as integer

.....
n = 6
.....

Redim a(n)

here array 'a' is declared without size. After execution begins we specify the size of the array with **redim.**

Example program

```
Dim a() As Integer
      Dim n
      Dim i
      n = Val(InputBox("Enter the element you want to add:"))
      ReDim a(n)
```

```
For i = 0 To n - 1
    a(i) = Val(InputBox("Enter element"))
Next

For i = 0 To n - 1
    MsgBox(a(i))
Next
```

Preserve Keyword : the preserve keyword helps to preserve the data in the existing array, when you resize it. If we don't specify preserve keyword then when we resize it its stored contents are lost. If we want to save the content then we need to specify **preserve** keyword.

Example program using preserve statement

```
Dim a() As Integer
        Dim n
        Dim s
        Dim i
        n = Val(InputBox("Enter the element you want to add:"))
        ReDim a(n)
        For i = 0 To n - 1
            a(i) = Val(InputBox("Enter element"))
        Next
        s = Val(InputBox("Enter element you want to add more:"))
        ReDim Preserve a(n + s - 1)
        For i = n To n + s - 1
            a(i) = Val(InputBox("Enter element"))
        Next
        For i = 0 To n + s - 1
            MsgBox(a(i))
        Next
```