**Arrays**

In visual basic, Arrays are useful to store multiple elements of the same data type at contiguous memory locations and arrays will allow us to store the fixed number of elements sequentially based on the predefined number of items.

An array can start storing the values from index 0. Suppose if we have an array with n elements, then it will start storing the elements from index 0 to n-1.

Arrays Declaration

In visual basic, Arrays can be declared by specifying the type of elements followed by the brackets () like as shown below.

Dim array\_name As [Data\_Type]();

Here, array\_name represents the name of an array and Data\_type will represent the data type of elements to store in an array.

Example

' Store only int values

Dim numbers As Integer()

' Store only string values

Dim names As String()

' Store only double values

Dim ranges As Double()

Arrays Initialization

In visual basic, Arrays can be initialized by creating an instance of an array with New keyword. By using the New keyword, we can declare and initialize an array at the same time based on our requirements.

' Declaring and Initializing an array with size of 5

Dim array As Integer() = New Integer(4) {}

' Defining and assigning an elements at the same time



Dim array2 As Integer() = New Integer(4) {1, 2, 3, 4, 5}

' Initialize with 5 elements will indicates the size of an array

Dim array3 As Integer() = New Integer() {1, 2, 3, 4, 5}



' Another way to initialize an array without size

Dim array4 As Integer() = {1, 2, 3, 4, 5}

' Declare an array without initialization

Dim array5 As Integer()

Visual Basic Accessing an Array Elements

In visual basic, we can access array elements by using for loop or foreach loop or with particular index numbers.

Following is the code snippet of accessing array elements by using particular index numbers.

Dim array As Integer() = New Integer(4) {1, 2, 3, 4, 5}

Dim a As Integer = array(1)

Dim b As Integer = array(4)

Following is the example of declaring, initializing and accessing array elements with particular index numbers in a visual basic programming language.

Dim array As Integer() = New Integer(4) {1, 2, 3, 4, 5}

Console.WriteLine(array(0))

Console.WriteLine(array(1))

Console.WriteLine(array(2))

Console.WriteLine(array(3))

Console.WriteLine(array(4))

Console.WriteLine("Press Enter Key to Exit..")

Console.ReadLine()

In the above example, we declared and initialized an array with 5 elements and we are accessing array elements using index values.

Visual Basic Access Array Elements with For Loop

In visual basic, by using for loop we can iterate through array elements and access the values of an array with length property.

Following is the example of accessing array elements using for loop in a visual basic programming language.

Dim array As Integer() = New Integer(4) {1, 2, 3, 4, 5}

For i As Integer = 0 To array.Length - 1

Console.WriteLine(array(i))

Next

Console.WriteLine("Press Enter Key to Exit..")

Console.ReadLine()

Visual Basic Access Array Elements with Foreach Loop

In visual basic, same as for loop we can use For Each loop to iterate through array elements and access the values of an array based on our requirements.

Following is the example of accessing array elements using For Each loop in a visual basic programming language.

Dim array As Integer() = New Integer(4) {1, 2, 3, 4, 5}

For Each i As Integer In array

Console.WriteLine(i)

Next

Console.WriteLine("Press Enter Key to Exit..")

Console.ReadLine()

Visual Basic Array Class

In visual basic, we have a class called Array and it will act as a base class for all the arrays in common language runtime (CLR). The Array class provides methods for creating, manipulating, searching and sorting arrays.

For example, by using Sort or Copy methods of Array class, we can sort the elements of an array and copy the elements of one array to another based on our requirements.

Dim arr As Integer() = New Integer(4) {21, 14, 27, 93, 25}

Console.WriteLine("---Initial Array Elements---")

For Each i As Integer In arr

Console.WriteLine(i)

Next

Array.Sort(arr)

Console.WriteLine("---Elements After Sort---")

For Each i As Integer In arr

Console.WriteLine(i)

Next

Array.Reverse(arr)

Console.WriteLine("---Elements After Reverse---")

For Each i As Integer In arr

Console.WriteLine(i)

Next

Console.WriteLine("Press Enter Key to Exit..")

Console.ReadLine()

