Arrays

Course Code : CS-127

Course Title : Object Oriented Programming

Semester : 2nd

C# Arrays

Like other programming languages, array in C# is a group of similar types of elements that have contiguous memory location. In C#, array is an *object* of base type **System.Array**. In C#, array index starts from 0. We can store only fixed set of elements in C# array.

C# Array Types

There are 3 types of arrays in C# programming:

- ✓ Single Dimensional Array
- ✓ Multidimensional Array
- ✓ Jagged Array

Declaring Arrays

To declare an array in C#, you can use the following syntax −

datatype[] arrayName;

where,

- ✓ *datatype* is used to specify the type of elements in the array.
- ✓ *[ ]* specifies the rank of the array. The rank specifies the size of the array.
- ✓ *arrayName* specifies the name of the array.

For example,

double[] balance;

Initializing an Array

- Declaring an array does not initialize the array in the memory. When the array variable is initialized, you can assign values to the array.

- We need to use the **new** keyword to create an instance of the array. For example,

```
double[] balance = new double[10];
```

C# Single Dimensional Array

- To create single dimensional array, we need to use square brackets [] after the type.

```
int[] arr = new int[5]; //creating array
```

- You cannot place square brackets after the identifier.

```
int arr[] = new int[5]; //compile time error
```

Declaration and Initialization at same time

- ✓ Declaring and Initializing an array with size of 5

```
int[] array = new int[5];
```

- ✓ Defining and assigning elements at the same time

```
int[] array2 = new int[5] {1,2,3,4,5};
```

- ✓ Initialize with 5 elements will indicates the size of an array

```
int[] array3 = new int[] { 1, 2, 3, 4, 5 };
```

Single Dimensional Array Example

```
1.  class Program
2.      {
3.          static void Main(string[] args)
4.          {
5.              int[] array = new int[5] { 1, 2, 3, 4, 5 };
6.              for (int i = 0; i < array.Length; i++)
7.              {
8.                  Console.WriteLine(array[i]);
9.              }
10.             Console.WriteLine("Press Enter Key to Exit..");
11.             Console.ReadLine();
12.         }
13.     }
```

Access Array Elements with Foreach Loop

- Same as for loop we can use the foreach loop to iterate through array elements and access the values of an array based on the requirements.

- Following is the example of accessing array elements using a foreach loop in c# programming language.

foreach loop

```
1.  class Program
2.      {
3.          static void Main(string[] args)
4.          {
5.              int[] array = new int[5] { 1, 2, 3, 4, 5 };
6.              foreach(int i in array)
7.              {
8.                  Console.WriteLine(i);
9.              }
10.             Console.WriteLine("Press Enter Key to Exit..");
11.             Console.ReadLine();
12.         }
13.     }
```

C# Array Class

- In c#, 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.

Example of using an Array class

```
using System;
namespace ConsoleApplication56
{
   class Program
   {
      static void Main(string[] args)
      {
         int[] numbers = new int[5] {33,21,54,67,54};
         Console.WriteLine(" Initial Array Elements  ");
         foreach(int a in numbers)
         {
            Console.WriteLine(a);
         }
```

```
        Array.Sort(numbers);
        Console.WriteLine(" Elements After Sort ");
        foreach (int a in numbers)
        {
           Console.WriteLine(a);
        }
        Array.Reverse(numbers);
        Console.WriteLine(" Elements After Reverse ");
        foreach (int a in numbers)
        {
           Console.WriteLine(a);
        }
        Console.Read();
           }
      }
}
```

C# Multidimensional Arrays

The multidimensional array is also known as rectangular arrays in C#. It can be two dimensional or three dimensional. The data is stored in tabular form (row * column) which is also known as matrix.

To create multidimensional array, we need to use comma inside the square brackets.

For example:

```
int[,] arr=new int[3,3];//declaration of 2D array

int[,,] arr=new int[3,3,3];//declaration of 3D array
```

C# Multidimensional Array Example1

Let's see a simple example of multidimensional array in C# which declares, initializes and traverse two dimensional array.

```
1.  using System;
2.  public class MultiArrayExample
3.  {
4.      public static void Main(string[] args)
5.      {
6.          int[,] arr=new int[3,3];//declaration of 2D array
7.          arr[0,1]=10;//initialization
8.          arr[1,2]=20;
9.          arr[2,0]=30;
10.
11.         //traversal
12.         for(int i=0;i<3;i++){
13.             for(int j=0;j<3;j++){
14.                 Console.Write(arr[i,j]+" ");
15.             }
16.             Console.WriteLine();//new line at each row
17.         }
18.     }
19. }
```

C# Multidimensional Array Example:

Declaration and initialization at same time:

```
int[,] arr = new int[3,3]
            {
              { 1, 2, 3 },
              { 4, 5, 6 },
              { 7, 8, 9 }
            };
```

C# Multidimensional Array Example2

Let's see a simple example of multidimensional array which initializes array at the time of declaration.

```
using System;
namespace ConsoleApplication52
{
   class Program
   {
      static void Main(string[] args)
      {
         int[,] numbers = new int[3, 3]
                          { // c0 c1 c2
                             {55,33,21},  //ro
                             {22,33,22},  //r1
                             {43,56,76}   //r2

                          };//declaration and initialization
```

```
for(int r = 0; r < 3; r++ )
        {
           for (int c = 0; c < 3; c++ )
           {
              if(numbers[r,c]%2==0)
              {
                 Console.WriteLine("Number is Even : "+numbers[r,c]);
              }
              else
              {
                 Console.WriteLine("Number is Odd : " + numbers[r, c]);
              }
           }
        }

        Console.ReadLine();
     }
  }
}
```

C# Jagged Arrays

- In C#, jagged array is also known as "array of arrays" because its elements are arrays. The element size of jagged array can be different.

Declaration of Jagged array

- Let's see an example to declare jagged array that has two elements.

```
int[][] arr = new int[2][];
```

Declaration of Jagged array

Let's see an example to declare jagged array that has two elements.

```
int[][] arr = new int[2][];
```

Initialization of Jagged array

Let's see an example to initialize jagged array. The size of elements can be different.

```
arr[0] = new int[4];
arr[1] = new int[6];
```

Initialization and filling elements in Jagged array

Let's see an example to initialize and fill elements in jagged array.

```
arr[0] = new int[4] { 11, 21, 56, 78 };
arr[1] = new int[6] { 42, 61, 37, 41, 59, 63 };
```

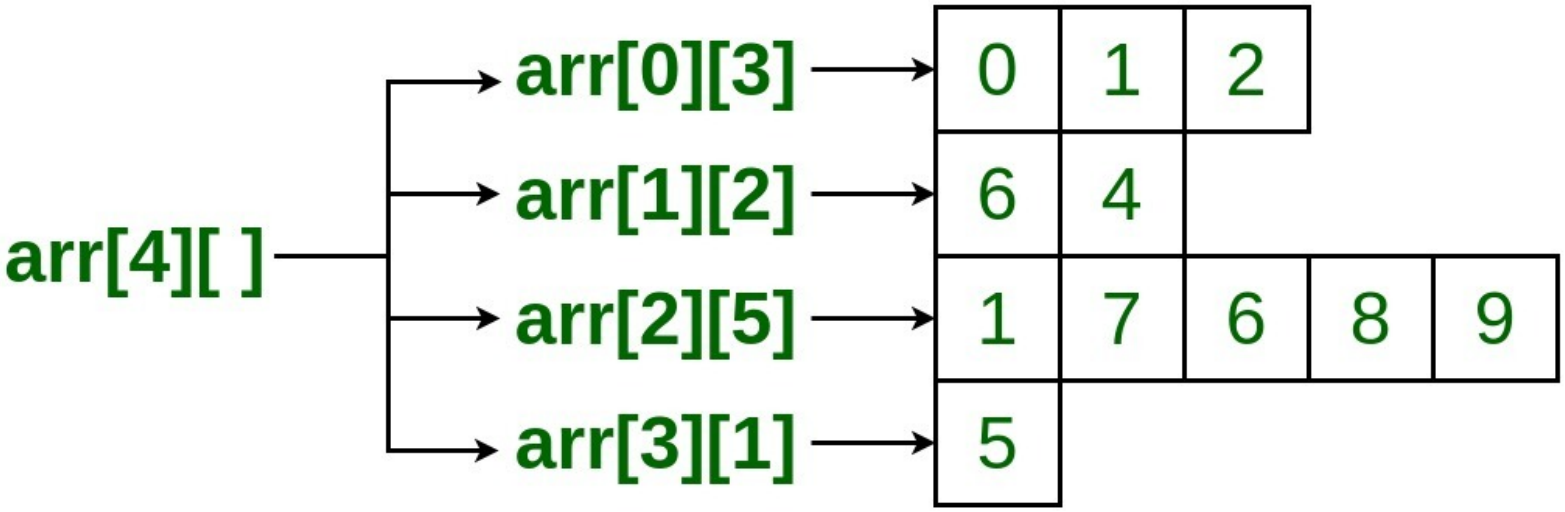
Rectangular array

[0,0]	[0,1]	[0,2]	[0,3]
[1,0]	[1,1]	[1,2]	[1,3]
[2,0]	[2,1]	[2,2]	[2,3]

Jagged array

[0].[0]	[0].[1]	[0].[2]	
[1].[0]	[1].[1]		
[2].[0]	[2].[1]	[2].[2]	[2].[3]

Jagged Array



C# Jagged Array Example

Let's see a simple example of jagged array in C# which declares, initializes and traverse jagged arrays.

```
using System;
namespace ConsoleApplication56
{
   class Program
   {
      static void Main(string[] args)
      {
         int[][] numbers = new int[2][];// jagged array declaration
         numbers[0] = new int[5] {33,43,65,76,87};//initialize the array
         numbers[1] = new int[6] { 33, 22, 44, 55, 76, 87 };
```

```
for (int i = 0; i < numbers.Length; i++)
        {
            for (int x = 0; x < numbers[i].Length; x++)
            {
                Console.WriteLine(numbers[i][x] + " ");
            }
        }
        Console.Read();
    }
}
```