Why do we need Arrays in programming?

As we know a primitive type variable such as int, double can hold only a single value at any given point in time. For example, int no = 10;. Here the variable "no" holds a value of 10. As per your business requirement, if you want to store 100 integer values, then you need to create 100 integer variables which is not a good programming approach as it will take lots of time as well as your code becomes bigger. So to overcome the above problems, Arrays in C# are introduced.

Types of Arrays in C#:

C# supports 2 types of arrays. They are as follows:

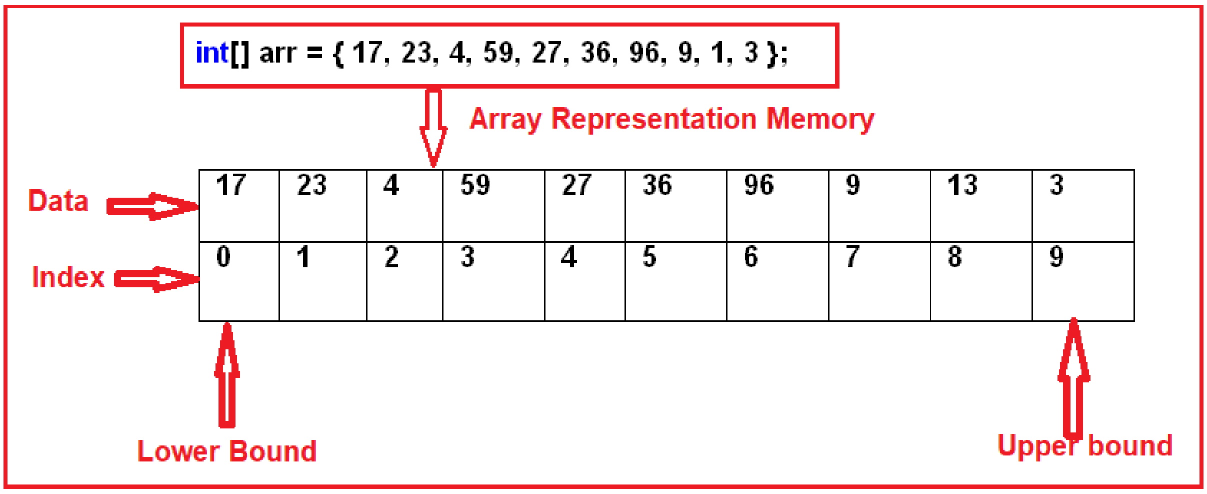
1. *Single dimensional array*
2. *Multi-dimensional array*

In the Single dimensional array, the data is arranged in the form of a row whereas in the Multi-dimensional arrays in C# the data is arranged in the form of rows and columns. Again the multi-dimensional arrays are of two types

1. *Jagged array: Whose rows and columns are not equal*
2. *Rectangular array: Whose rows and columns are equal*

Memory Representation of Arrays in C#:

Please have a look at the following diagram:


int[] arr = { 17, 23, 4, 59, 27, 36, 96, 9, 1, 3 };
Array Representation Memory
Data
Index
17 23 4 59 27 36 96 9 13 3
0 1 2 3 4 5 6 7 8 9
Lower Bound
Upper bound

In C#, the arrays can be declared as fixed-length or dynamic. The Fixed length array means we can store a fixed number of elements while in the case of the dynamic array, the size of the array automatically increases as we add new items into the array.

One Dimensional Array in C# with Examples:

The array which stores the data in the form of rows in a sequential order is called a one-dimensional array in C#. The syntax for creating a one-dimensional array in C# is given below.

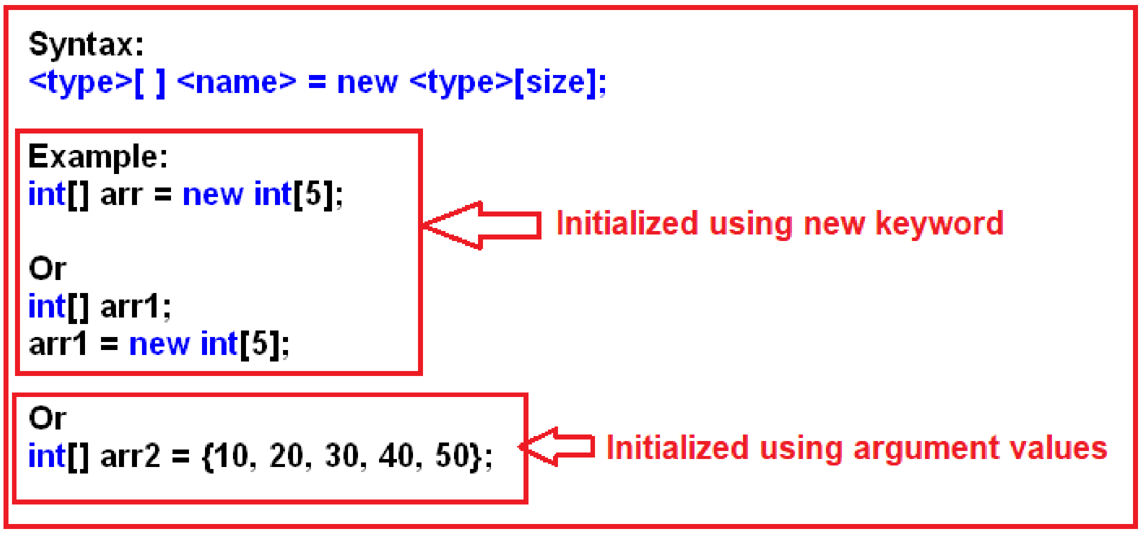
EXAMPLE

Table of 2

```
 class Program

   {

         static void Main(string[] args)

         {
            int j, n;



            Console.Write("\n\n");

            Console.Write("Display the multiplication table:\n");

            Console.Write("-----------------------------------");

            Console.Write("\n\n");



            Console.Write("Input the number (Table to be calculated) : ");

            n = Convert.ToInt32(Console.ReadLine());

            Console.Write("\n");

            for (j = 1; j <= 10; j++)

            {

                Console.Write("{0} X {1} = {2} \n", n, j, n * j);

            }

        }

    }
```

The Array class in C# is not a part of the System.Collections namespace. It is a part of the System namespace. But still, we considered it as a collection because it is Implements the IList interface. The Array class provides the following methods and properties:

1. *Sort(<array>): Sorting the array elements*
2. *Reverse (<array>): Reversing the array elements*
3. *Copy (src, dest, n): Copying some of the elements or all elements from the old array to the new array*
4. *GetLength(int): A 32-bit integer that represents the number of elements in the specified dimension.*
5. *Length: It Returns the total number of elements in all the dimensions of the Array; zero if there are no elements in the array.*



```
Syntax:
<type>[ ] <name> = new <type>[size];

Example:
int[] arr = new int[5];

Or
int[] arr1;
arr1 = new int[5];

Or
int[] arr2 = {10, 20, 30, 40, 50};
```

Initialized using new keyword
Initialized using argument values

<u>What is a Two-Dimensional Array in C#?</u>

The arrays which store the elements in the form of rows and columns are called Two-Dimensional Array in C#. The two-dimensional array which is also called multidimensional array is of two types in C#. They are as follows

1. *Rectangular array: The array whose rows and columns are equal are called a rectangular array*
2. *Jagged array: The array whose rows and columns are not equal are called a jagged array*

<u>EXAMPLE</u>

```
class Program

  {

      static void Main(string[] args)
```

```
{
    int[] arr = { 14, 15, 23, 4, 56, 67, 43, 23, 28, 12 };
    Console.WriteLine("the arre cotain the below element");
    for(int i=0;i<arr.Length;i++)
    {
        Console.WriteLine(arr[i] + "");


    }
    Console.WriteLine();
   // Array.Copy(arr, brr, 5);


    int[] brr = new int[10];
    Console.WriteLine("the new array element are ");
    Array.Copy(arr, brr, 5);
    Console.WriteLine("the array element after sorting");
    foreach(int i in arr)
    {
        Console.WriteLine(i + "");


    }
```

```
          // Array.Copy(arr, brr, 5);


           foreach (int i in brr)

           {

               Console.WriteLine(i + "");

           }




       }

   }

}
```

<u>*What are the Array and their disadvantages in C#?*</u>

In simple words, we can say that the Arrays in C# are the simple data structure that is used to store similar types of data items in sequential order. Although the arrays in c# are commonly used, they have some limitations.

For example, you need to specify the array's size while creating the array. If at execution time, you want to modify it that means if you want to increase or decrease the size of an array, then you need to do it manually by creating a new array or by using the Array class's Resize method, which internally creates a new array and copies the existing array element into the new array.

<u>Following are the limitations of Array in C#:</u>

1. *The array size is fixed. Once the array is created we can never increase the size of an array. If we want then we can do it manually by creating a new array and copy the old array elements into the new array or by using the Array class Resize method which will do the same thing means to create a new array and copy the old array elements into the new array and then destroy the old array.*
2. *We can never insert an element into the middle of an array*
3. *Deleting or removing elements from the middle of the array.*

To overcome the above problems, the Collections are introduced in C# 1.0.

<u>What is a Collection in C#?</u>

The Collections in C# are a set of predefined classes that are present in the System.Collections namespace that provides greater capabilities than the traditional arrays. The collections in C# are reusable, more powerful, more efficient and most importantly they have been designed and tested to ensure quality and performance.

So in simple words, we can say a Collection in C# is a dynamic array. That means the collections in C# have the capability of storing multiple values but with the following features.

1. *Size can be increased dynamically.*
2. *We can insert an element into the middle of a collection.*
3. *It also provides the facility to remove or delete elements from the middle of a collection.*

The collection which is from .Net framework 1.0 is called simply collections or Non-Generic collections in C#. These collection classes are present inside the

System.Collections namespace. The example includes are Stack, Queue, LinkedList, SortedList, ArrayList, HashTable, etc.

<u>What is ArrayList in C#?</u>

The ArrayList in C# is a collection class that works like an array but provides the facilities such as dynamic resizing, adding, and deleting elements from the middle of a collection. It implements the System.Collections.IList interface using an array whose size is dynamically increased as required.

<u>Methods and Properties of ArrayList Collection class in C#:</u>

The following are the methods and properties provided by the ArrayList collection class in C#.

1. *Add(object value): This method is used to add an object to the end of the collection.*
2. *Remove(object obj): This method is used to remove the first occurrence of a specific object from the collection.*
3. *RemoveAt(int index): This method takes the index position of the elements and removes that element from the collection.*
4. *Insert(int index, Object value): This method is used to inserts an element into the collection at the specified index.*
5. *Capacity: This property gives you the capacity of the collection means how many elements you can insert into the collection.*

<u>What is the difference between an array and an Array List in C#?</u>

This is one of the frequently asked interview questions in C#. So let us discuss the difference between an array and ArrayList.

Array:

1. *Fixed Length*
2. *Cannot insert into the middle*
3. *Cannot delete from middle*

ArrayList:

1. *Variable Length*
2. *Can insert an element into the middle of the collection*
3. *Can delete element from the middle of the collection*