Technical Institute of Administration

Management Information System

Computer Programming

7. Array

Lecturer:

Sipan M. Hameed

www.sipan.dev

2024-2025

1. Table of Contents

7. Arrays

You can store multiple variables of the same type in an array data structure. You declare an array by specifying the type of its elements. If you want the array to store elements of any type, you can specify object as its type. In the unified type system of C#, all types, predefined and user-defined, reference types and value types, inherit directly or indirectly from Object.

7.1 C# Break and Continue

7.1.1 C# Break

You have already seen the break statement used in an earlier chapter of this tutorial. It was used to "jump out" of a switch statement.

The break statement can also be used to jump out of a loop.

This example jumps out of the loop when i is equal to 4:

Example:

```
using System;
public class Ex71
{
    public static void Main()
    {
        for (int i = 0; i < 10; i++)
        {
            if (i == 4)
            {
                break;
            }
            Console.WriteLine(i);
        }
    }
}
```

The output:

```
0
1
2
3
```

7.1.2 C# Continue

The continue statement breaks one iteration (in the loop), if a specified condition occurs, and continues with the next iteration in the loop.

This example skips the value of 4:

Example:

```
using System;
public class Ex72
{
    public static void Main()
    {
        for (int i = 0; i < 10; i++)
        {
            if (i == 4)
            {
                continue;
            }
            Console.WriteLine(i);
        }

    }
}
```

Output:

```
0
1
2
3
5
6
7
8
9
```

7.2 Arrays (C# Programming Guide)

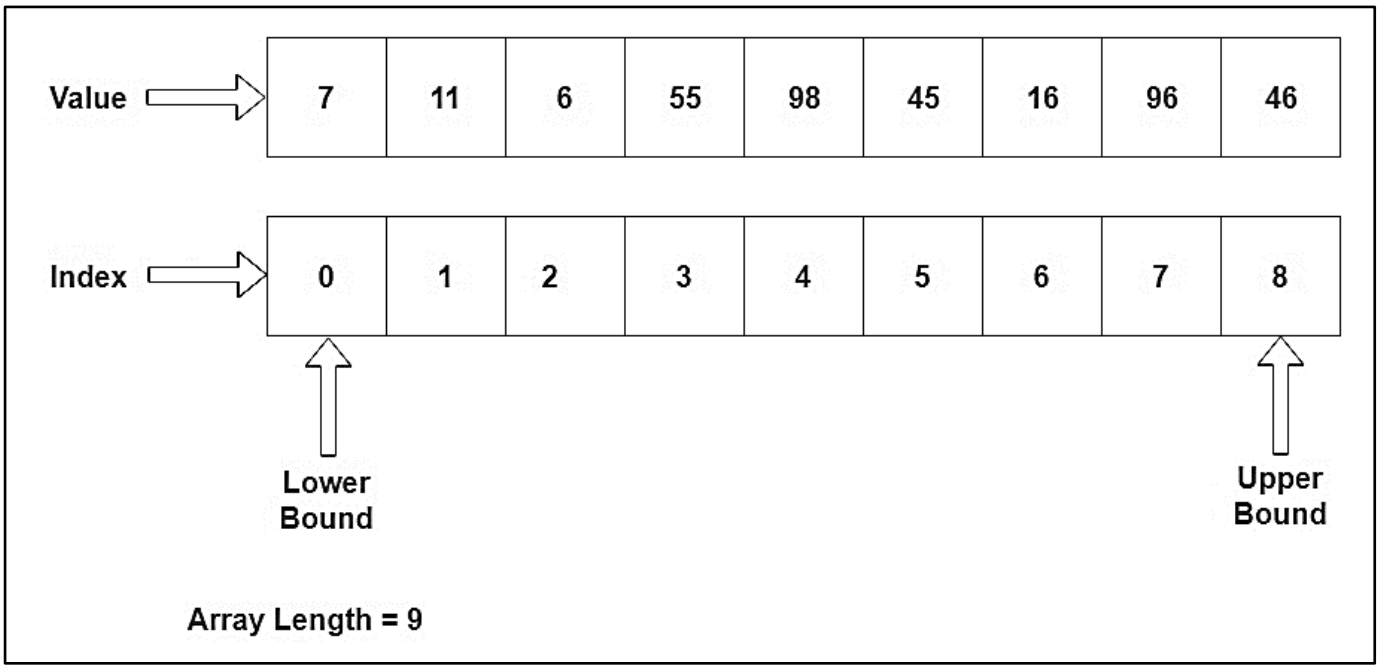
An array is a group of like-typed variables that are referred to by a common name. And each data item is called an element of the array. The data types of the elements may be any valid data type like char, int, float, etc. and the elements are stored in a contiguous location. Length of the array specifies the number of elements present in the array. In C# the allocation of memory for the arrays is done dynamically. And arrays are kind of objects, therefore it is easy to find their size using the predefined functions. The variables in the array are ordered and each has an index beginning from 0. Arrays in C# work differently than they do in C/C++.

7.2.1 Important Points to Remember About Arrays in C#

- In C#, all arrays are dynamically allocated.
- An array can be **single-dimensional**, **multidimensional** or **jagged**.
- Since arrays are objects in C#, we can find their length using member length. This is different from C/C++ where we find length using **sizeof** operator.
- A C# array variable can also be declared like other variables with [] after the data type.
- The variables in the array are ordered and each has an index beginning from 0.
- C# array is an object of base type System.Array.
- Default values of numeric array and reference type elements are set to be respectively zero and null.
- A jagged array elements are reference types and are initialized to null.
- Array elements can be of any type, including an array type.
- Array types are reference types which are derived from the abstract base type Array. These types implement IEnumerable and for it, they use foreach iteration on all arrays in C#.

The array has can contains primitives data types as well as objects of a class depending on the definition of an array. Whenever use primitives data types, the actual values have to be stored in contiguous memory locations. In the case of objects of a class, the actual objects are stored in the heap segment.

The following figure shows how array stores values sequentially:



Explanation:

The index is starting from 0, which stores value. we can also store a fixed number of values in an array. Array index is to be increased by 1 in sequence whenever it's not reached the array size.

7.2.2 Array Declaration

Syntax :

```
< Data Type > [ ] < Name_Array >
```

7.2.2.1 Example :

```
int[] x;                // can store int values
string[] s;             // can store string values
double[] d;             // can store double values
Student[] stud1;   // can store instances of Student class which is custom class
```

7.2.3 Array Initialization

As said earlier, an array is a reference type so the new keyword used to create an instance of the array. We can assign initialize individual array elements, with the help of the index.

Syntax :

```
type [ ] < Name_Array > = new < datatype > [size];
```

Examples : To Show Different ways for the Array Declaration and Initialization

7.2.3.1 Example 1 :

```
// defining array with size 5.
// But not assigns values
int[] intArray1 = new int[5];
```

The above statement declares & initializes int type array that can store five int values. The array size is specified in square brackets([]).

7.2.3.2 Example 2 :

```
// defining array with size 5 and assigning
// values at the same time
int[] intArray2 = new int[5]{1, 2, 3, 4, 5};
```

The above statement is the same as, but it assigns values to each index in {}.

7.2.3.3 Example 3 :

```
// defining array with 5 elements which
// indicates the size of an array
int[] intArray3 = {1, 2, 3, 4, 5};
```

In the above statement, the value of the array is directly initialized without taking its size. So, array size will automatically be the number of values which is directly taken.

7.2.4 Initialization of an Array after Declaration

Arrays can be initialized after the declaration. It is not necessary to declare and initialize at the same time using the new keyword. However, Initializing an Array after the declaration, it must be initialized with the new keyword. It can't be initialized by only assigning values.

Example :

```
// Declaration of the array
string[] str1, str2;
// Initialization of array
str1 = new string[5]{ "Element 1", "Element 2", "Element 3", "Element 4", "Element 5" };
str2 = new string[5]{ "Element 1", "Element 2", "Element 3", "Element 4", "Element 5" };
```

7.2.5 Note:

Initialization without giving size is not valid in C#. It will give a compile-time error.

Example: Wrong Declaration for initializing an array

```
// compile-time error: must give size of an array
int[] intArray = new int[];

// error : wrong initialization of an array
string[] str1;
str1 = {"Element 1", "Element 2", "Element 3", "Element 4" };
```

7.2.6 Accessing Array Elements

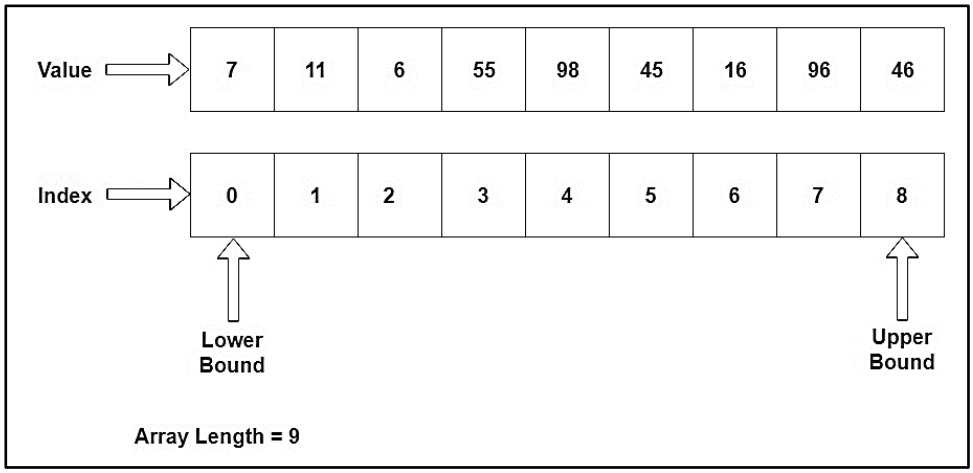
At the time of initialization, we can assign the value. But, we can also assign the value of the array using its index randomly after the declaration and initialization. We can access an array value through indexing, placed index of the element within square brackets with the array name

Example :

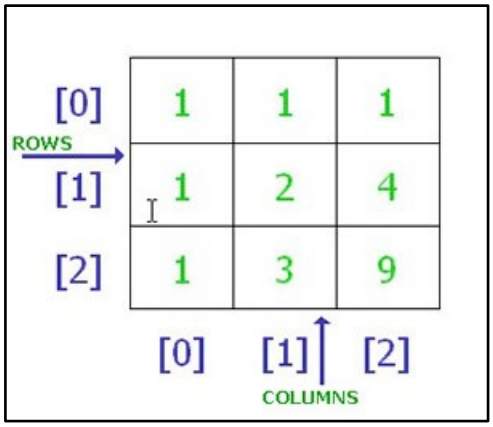
```
//declares & initializes int type array
int[] intArray = new int[5];

// assign the value 10 in array on its index 0
intArray[0] = 10;

// assign the value 30 in array on its index 2
intArray[2] = 30;

// assign the value 20 in array on its index 1
intArray[1] = 20;

// assign the value 50 in array on its index 4
intArray[4] = 50;

// assign the value 40 in array on its index 3
intArray[3] = 40;

// Accessing array elements using index
intArray[0];  //returns 10
intArray[2];  //returns 30
```

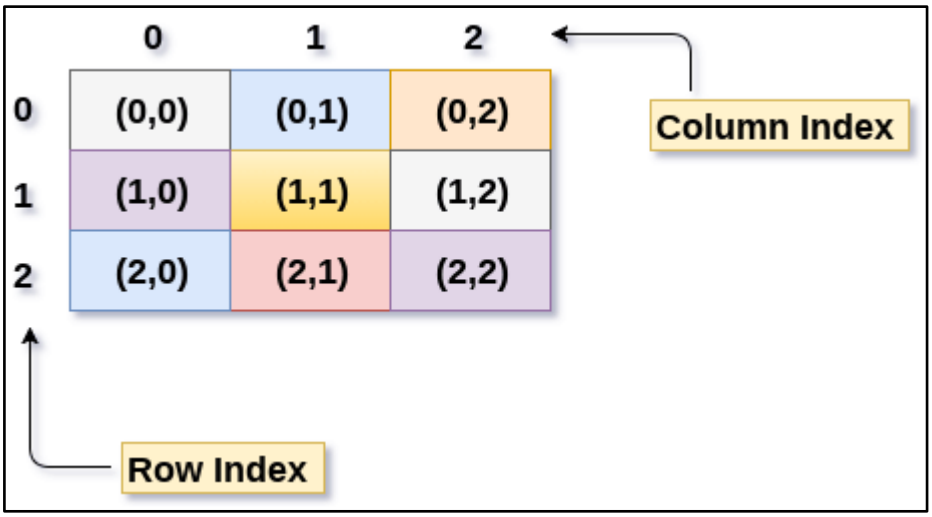
7.3 Array dimensions visualization

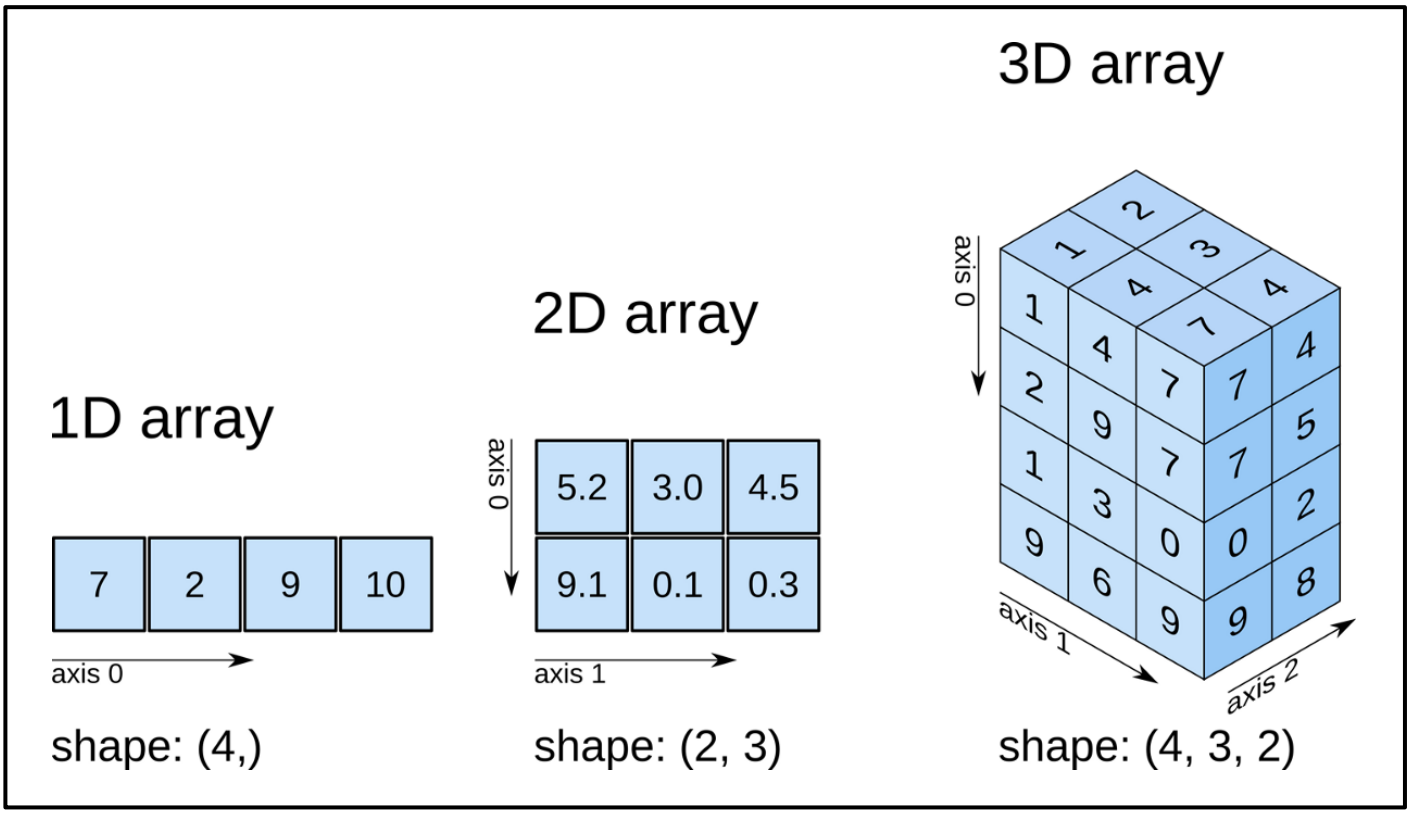
7.3.1 one dimension array

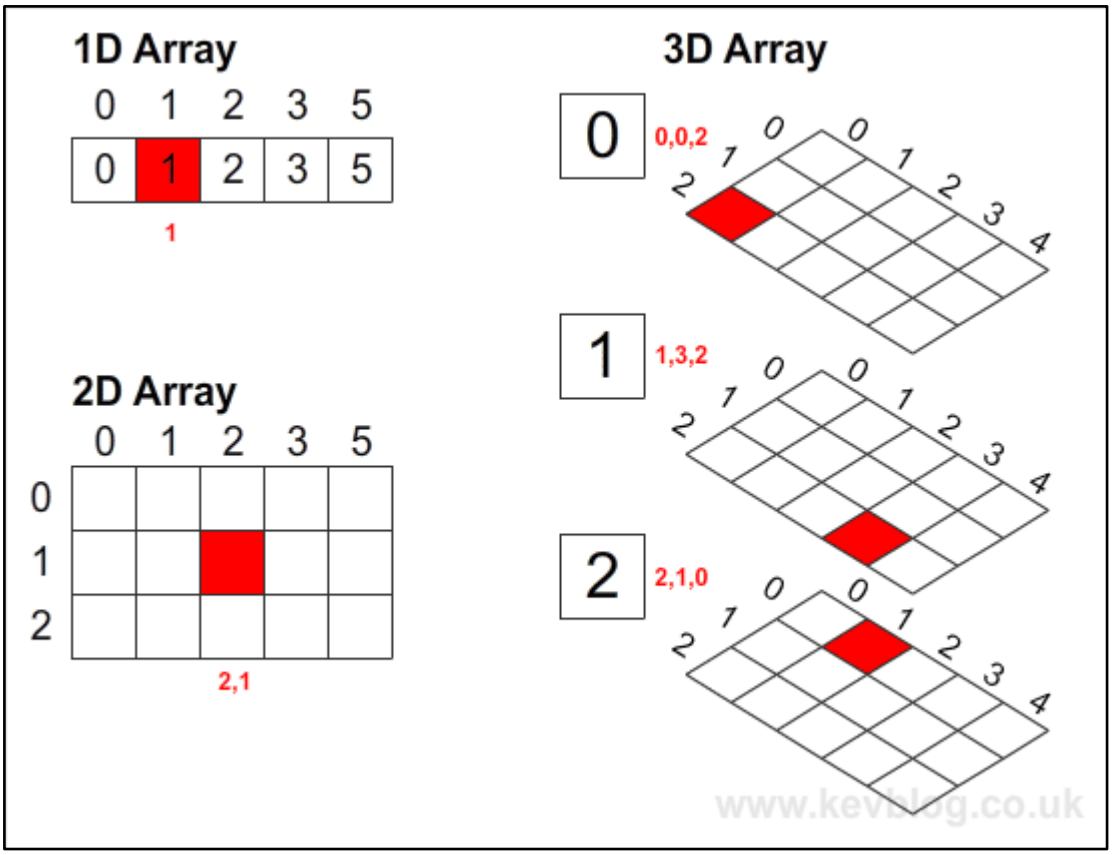


7.3.2 2d array (rows and columns)



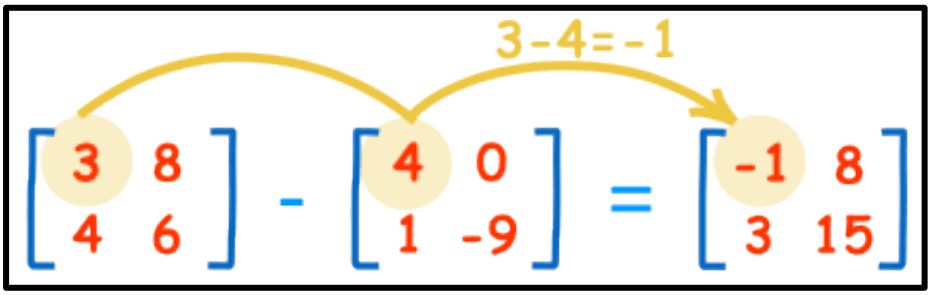


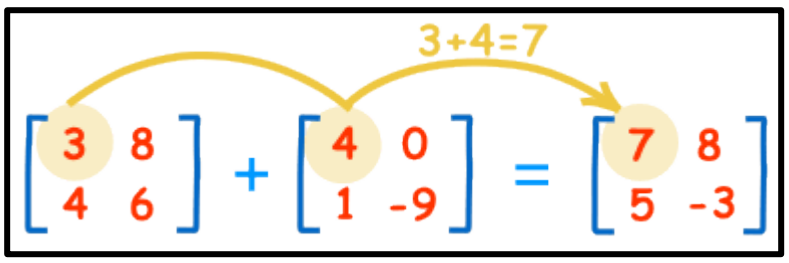

3D array
1D array
2D array
7
2
9
10
axis 0
5.2
3.0
4.5
9.1
0.1
0.3
axis 0
axis 1
axis 0
axis 1
axis 2
shape: (4,)
shape: (2, 3)
shape: (4, 3, 2)

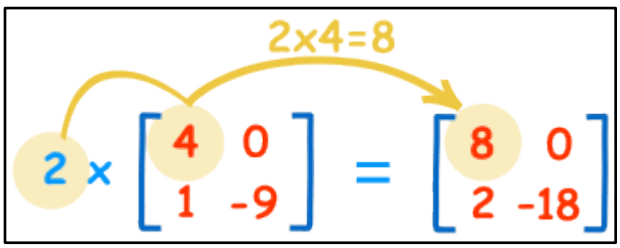

1D Array
0 1 2 3 5
0 1 2 3 5
1
2D Array
0 1 2 3 5
0
1
2
2,1
3D Array
0
0,0,2
1
1,3,2
2
2,1,0
www.kevsblog.co.uk

7.3.3 matrix

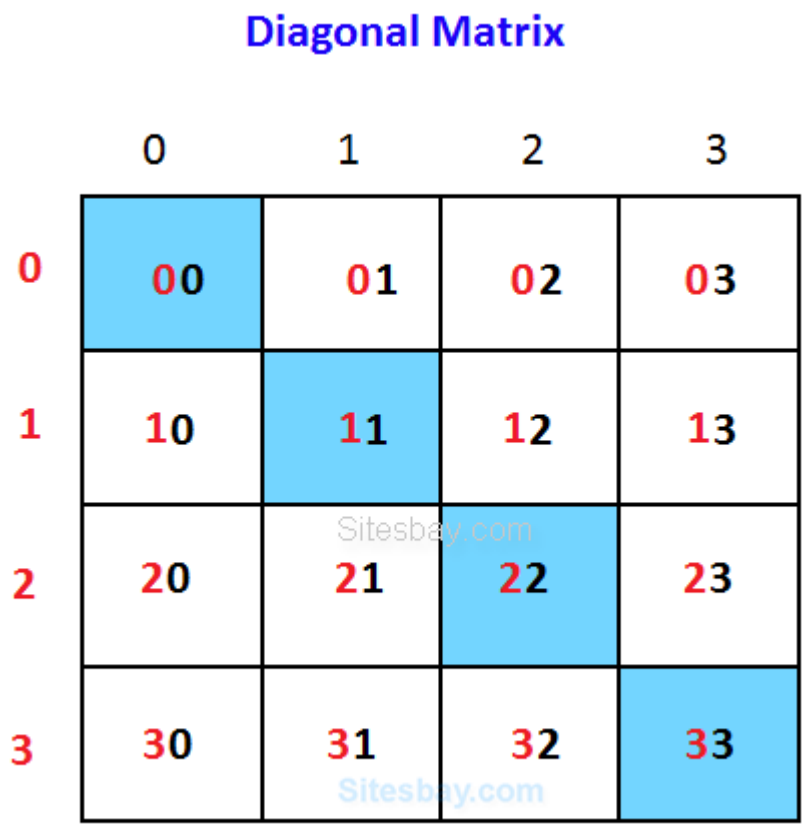
	Col 1	Col 2	Col 3	Col 4
Row 1	0, 0	0, 1	0, 2	0, 3
Row 2	1, 0	1, 1	1, 2	1, 3
Row 3	2, 0	2, 1	2, 2	2, 3
Row 4	3, 0	3, 1	3, 2	3, 3

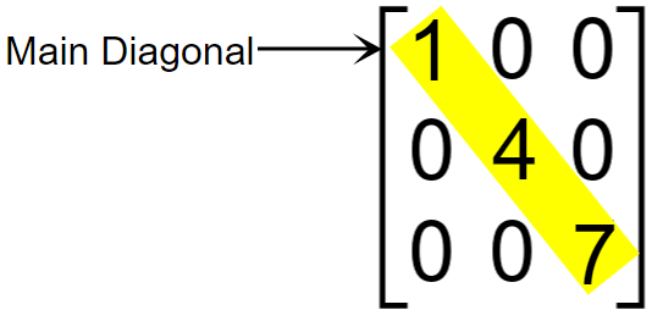






7.3.4 Matrix Diagonals





1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Main

1st Diagonal: 1, 6, 11, 16

Second

2nd Diagonal: 13, 10, 7, 4

7.4 Array dimensions

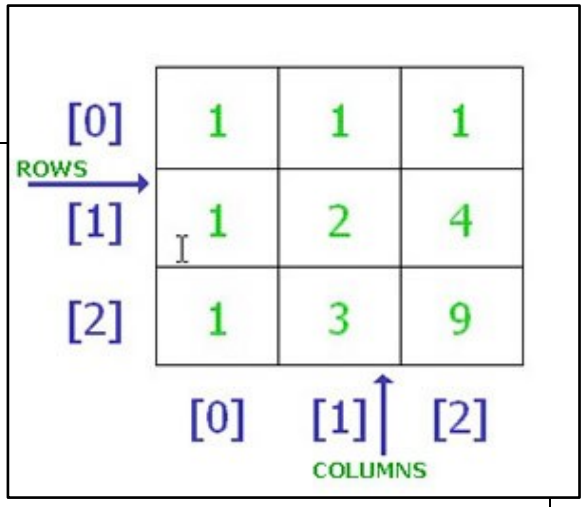
7.4.1 Examples-1

The following example creates single-dimensional, multidimensional, and jagged arrays:

```
class TestArraysClass
{
    static void Main()
    {
        // Declare a single-dimensional array of 5 integers.
        int[] array1 = new int[5];

        // Declare and set array element values.
        int[] array2 = new int[] { 1, 3, 5, 7, 9 };

        // Alternative syntax.
        int[] array3 = { 1, 2, 3, 4, 5, 6 };

        // Declare a two dimensional array.
        int[,] array2d = new int[2, 3];

        // Declare and set array element values.
        int[,] aa2d = { { 1, 2, 3 }, { 4, 5, 6 } };

        // Declare and set array element values.
        int[,] arrr = new int[2,3]{ { 1, 2, 3 }, { 4, 5, 6 } };

        // Declare a jagged array.
        int[][] jaggedArray = new int[6][];

        // Set the values of the first array in the jagged array structure.
        jaggedArray[0] = new int[4] { 1, 2, 3, 4 };
    }
}
```

7.4.2 Example 2:

Create the following array:

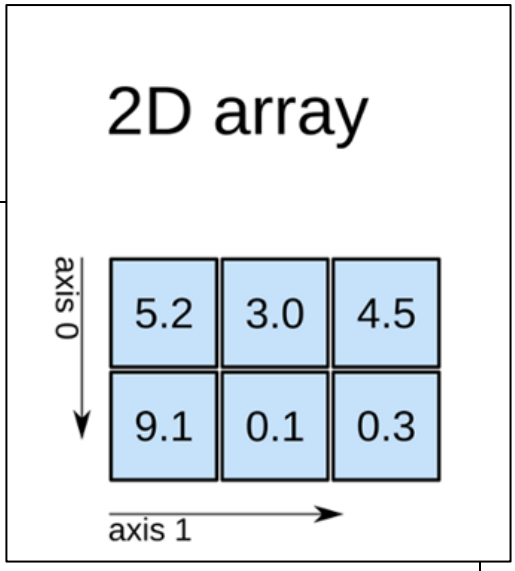



```
using System;
class arrm2
{
    static void Main()
    {
        int[,] arr = new int[3, 3];

        for (int i = 0; i < 3; i++)
        {
            for (int j = 0; j < 3; j++)
            {
                arr[i, j] = Convert.ToInt32(Console.ReadLine());
            }
        }

         Console.WriteLine("output:");

        for (int i = 0; i < 3; i++)
        {
            for (int j = 0; j < 3; j++)
            {
                Console.Write(arr[i,j]+"   ");
            }
            Console.WriteLine();
        }

    }
}
```

Output:

```
1
1
1
1
2
4
1
3
9
output:
1   1   1
1   2   4
1   3   9
```

7.4.3 Example 3:



Create the following array:

```
using System;
class arrm3
{
    static void Main()
    {
        double[,] arr = new double[2, 3];

        for (int i = 0; i < 2; i++)
        {
            for (int j = 0; j < 3; j++)
            {
                arr[i, j] = Convert.ToDouble(Console.ReadLine());
            }
        }

        Console.WriteLine("output:");

        for (int i = 0; i < 2; i++)
        {
            for (int j = 0; j < 3; j++)
            {
                Console.Write(arr[i,j]+"   ");
            }
            Console.WriteLine();
        }

    }
}
```

Output:

```
5.2
3.0
4.5
9.1
0.1
0.3
output:
5.2   3   4.5
9.1   0.1   0.3
```

7.4.4 Example 4:

Create the following array:

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

```
using System;
class arrm4
{
    static void Main()
    {
        int[,] arr = new int[5, 5];
        int m = 1;

        for (int i = 0; i < 5; i++)
        {
            for (int j = 0; j < 5; j++)
            {
                arr[i, j] = m;
                m++;
            }
        }

        Console.WriteLine("output: ");

        for (int i = 0; i < 5; i++)
        {
            for (int j = 0; j < 5; j++)
            {
                Console.Write(arr[i,j]+"     ");
            }
            Console.WriteLine();
        }

    }
}
```

output:

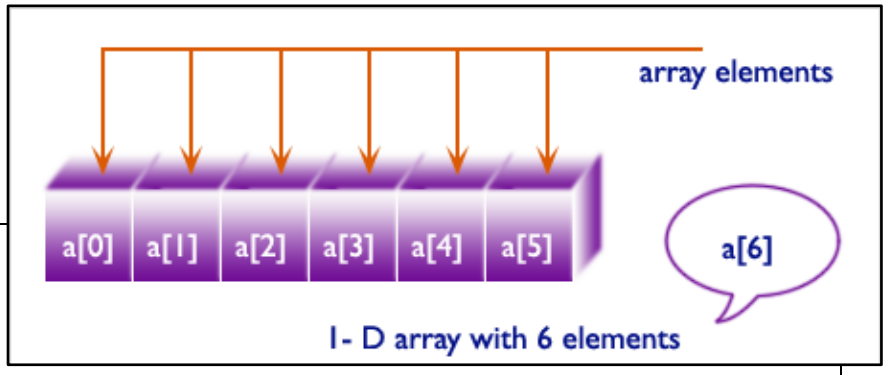
```
1     2     3     4     5
6     7     8     9     10
11     12     13     14     15
16     17     18     19     20
21     22     23     24     25
```

7.5 Practical Examples

7.5.1 Array: Exercise-1 with Solution

Write a program in C# Sharp to store 6 elements in an array and print it.

C# Sharp: Read and Print elements of an array



Sample Solution:-

C# Sharp Code:

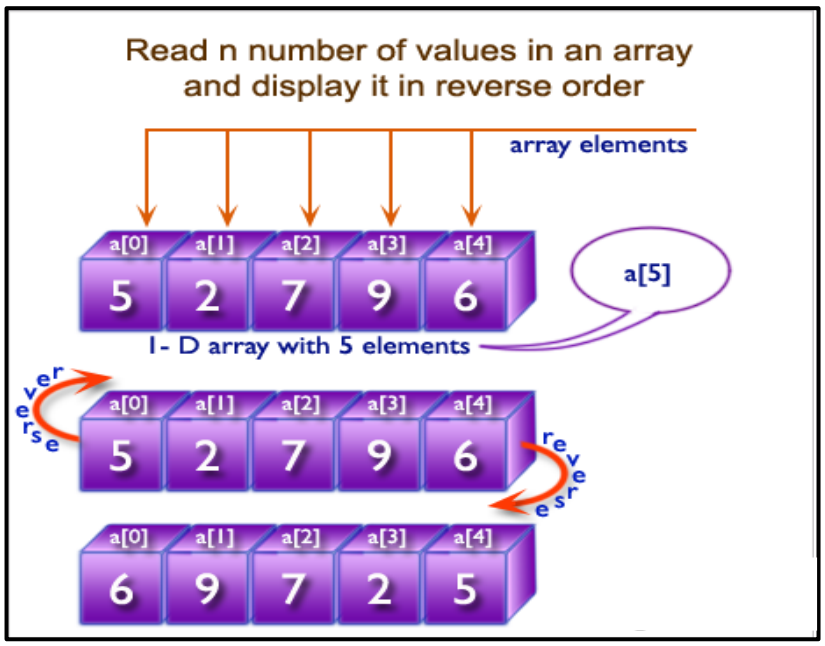
```
using System;
class arrex1
{
    static void Main()
    {
        int[] arr = new int[6];

        for (int i = 0; i < 6; i++)
        {
            arr[i] = Convert.ToInt32(Console.ReadLine());
        }

        Console.WriteLine("output: ");

        for (int i = 0; i < 6; i++)
        {
            Console.Write(arr[i]+ "   ");
        }

    }
}
```

Output:

```
33
5
67
89
1
4
output:
33   5   67   89   1   4   Press any key to continue . . .
```

7.5.2 Array: Exercise-2 with Solution

Write a program in C# Sharp to read n number of values in an array with **n** length and display it in reverse order.



Sample Solution:-

C# Sharp Code:

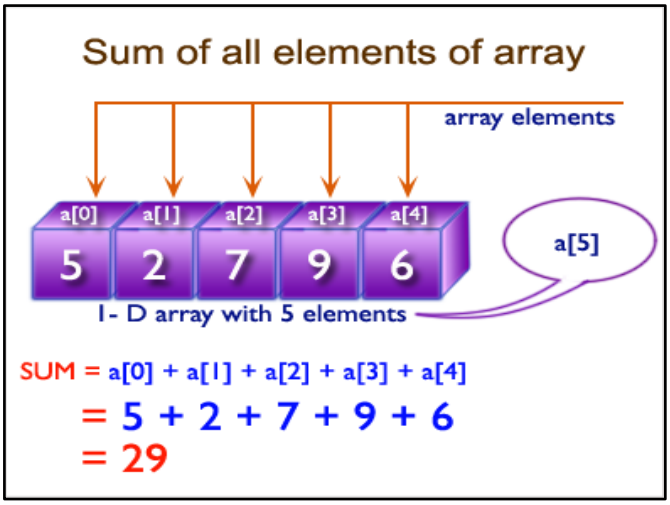
```
using System;



class arrex2
{
    static void Main()
    {
        Console.WriteLine("enter array size: ");
        int n = Convert.ToInt32(Console.ReadLine());

        int[] arr = new int[n];

        Console.WriteLine("enter array:");
        for (int i = 0; i < n; i++)
        {
            arr[i] = Convert.ToInt32(Console.ReadLine());
        }

        Console.WriteLine("output: ");

        for (int i = n-1; i >= 0; i--)
        {
            Console.Write(arr[i]+ "    ");
        }
        Console.WriteLine();
    }
}
```

Output:

```
enter array size:
5
enter array:
88
76
45
987
432
output:
432    987    45    76    88
```

7.5.3 Array: Exercise-3 with Solution

Write a program in C# Sharp to find the sum of all elements of array.



Sample Solution:-

C# Sharp Code:

```
using System;



class arrex3
{
    static void Main()
    {
        Console.WriteLine("enter array size: ");
        int n = Convert.ToInt32(Console.ReadLine());
        int sum = 0;
        int[] arr = new int[n];

        Console.WriteLine("enter array:");
        for (int i = 0; i < n; i++)
        {
            arr[i] = Convert.ToInt32(Console.ReadLine());
        }

        Console.WriteLine("output: ");

        for (int i = n-1; i >=0; i--)
        {
            sum = sum + arr[i];
        }
        Console.WriteLine("sum =  "+sum);
    }
}
```

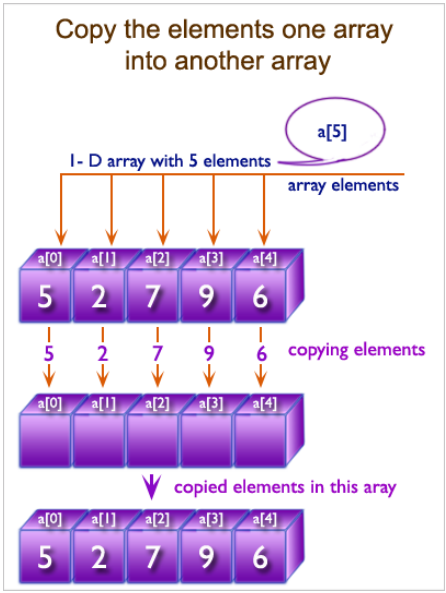
Output:

```
enter array size:
5
enter array:
5
2
7
9
6
output:
sum =  29
```

7.5.4 Array: Exercise-4 with Solution

Write a program in C# Sharp to copy the elements one array into another array.

Sample Solution:-



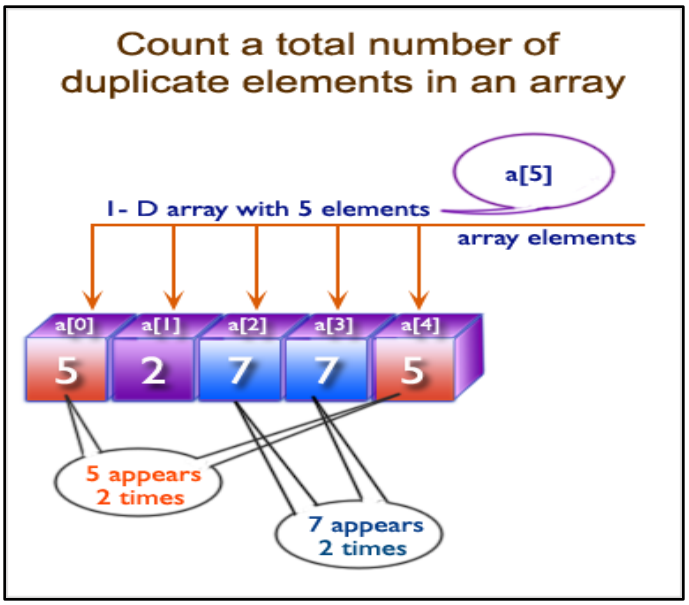
C# Sharp Code:

```
using System;

class arrex4
{
    static void Main()
    {
        Console.WriteLine("enter array size: ");
        int n = Convert.ToInt32(Console.ReadLine());
        int[] arr1 = new int[n];
        int[] arr2 = new int[n];
        Console.WriteLine("enter array:");
        for (int i = 0; i < n; i++)
        {
            arr1[i] = Convert.ToInt32(Console.ReadLine());
        }

        for (int i = n - 1; i >= 0; i--)
        {
            arr2[i] = arr1[i];
        }

        Console.WriteLine("output: ");

        for (int i = n-1; i >=0; i--)
        {
            Console.Write(arr2[i]+"   " );
        }
        Console.WriteLine();
    }
}
```

Output:

```
enter array size:
5
enter array:
5
9
456
235
88
output:
88   235   456   9   5
```

7.5.5 Array: Exercise-5 with Solution



Write a program in C# Sharp to count a total number of duplicate elements in an array.

Sample Solution: -

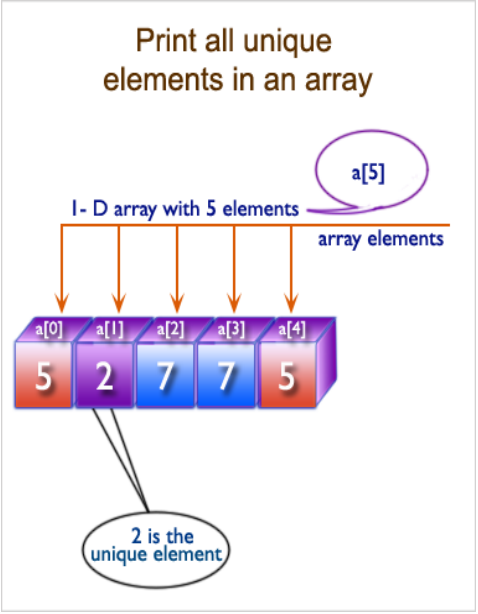
C# Sharp Code:

```
using System;



class arrex5
{
    static void Main()
    {

        Console.WriteLine("enter array size: ");
        int n = Convert.ToInt32(Console.ReadLine());
        int[] arr1 = new int[n];
        int[] arr2 = new int[n];
        int[] arr3 = new int[n];
        Console.WriteLine("enter array:");
        for (int i = 0; i < n; i++)
        {
            arr1[i] = Convert.ToInt32(Console.ReadLine());
        }
        for (int i = 0; i  <n; i++)
        {
            arr2[i] = arr1[i];
        }
        Console.WriteLine("output: ");
        for (int i = 0; i < n; i++)
        {
            for (int j = 0; j < n; j++)
            {
                if (arr2[i]==arr1[j])
                {
                    arr3[i]++;
                }
            }
        }

        for (int i = 0; i < n; i++)
        {
            Console.WriteLine(arr2[i]+ " : " +arr3[i]);
        }
        Console.WriteLine();
    }
}
```

Output:

```
enter array size:
6
enter array:
7
8
55
4
55
7
output:
7 : 2
8 : 1
55 : 2
4 : 1
55 : 2
7 : 2
```

7.5.6 Array: Exercise-6 with Solution



Write a program in C# Sharp to print all unique elements in an array.

Sample Solution:-

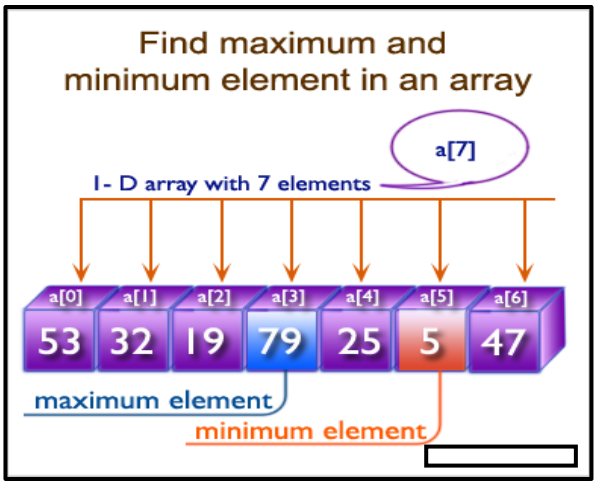
C# Sharp Code:

```
using System;
class arrex6
{
    static void Main()
    {
        Console.WriteLine("enter array size: ");
        int n = Convert.ToInt32(Console.ReadLine());
        int[] arr1 = new int[n];
        int[] arr2 = new int[n];
        int[] arr3 = new int[n];
        Console.WriteLine("enter array:");
        for (int i = 0; i < n; i++)
        {
            arr1[i] = Convert.ToInt32(Console.ReadLine());
        }
        for (int i = 0; i  <n; i++)
        {
            arr2[i] = arr1[i];
        }

        Console.WriteLine("output: ");

        for (int i = 0; i < n; i++)
        {
            for (int j = 0; j < n; j++)
            {
                if (arr2[i]==arr1[j])
                {
                    arr3[i]++;
                }
            }
        }

        for (int i = 0; i < n; i++)
        {
            if (arr3[i] == 1)
            {
                Console.WriteLine(arr2[i] + "  ");
            }
        }
        Console.WriteLine();
    }
}
```

Output:

```
enter array size:
5
enter array:
5
2
7
7
5
output:
2
```

7.5.7 Array: Exercise-7 with Solution



Write a program in C# Sharp to find maximum and minimum element in an array.

Sample Solution:-

C# Sharp Code:

```
using System;

internal class arrex7
{
    private static void Main()
    {
        Console.WriteLine("enter array size: ");
        int n = Convert.ToInt32(Console.ReadLine());
        int[] arr1 = new int[n];

        Console.WriteLine("enter array:");
        for (int i = 0; i < n; i++)
        {
            arr1[i] = Convert.ToInt32(Console.ReadLine());
        }
        int mx = arr1[0];
        int mn = arr1[0];
        Console.WriteLine("output: ");

        for (int i = 0; i < n; i++)
        {
            if (arr1[i] > mx)
            {
                mx = arr1[i];
            }
            if (arr1[i] < mn)
            {
                mn = arr1[i];
            }
        }

        Console.WriteLine(mx + "    " + mn);
    }
}
```

Output:

```
enter array size:
6
enter array:
3
44
7
87
1
3
output:
max = 87    min = 1
```

7.5.8 Array: Exercise-8 with Solution

Write a program in C# Sharp to index of matrix (2d array).**M** x **N**

Sample Solution:-

C# Sharp Code:

	Col 1	Col 2	Col 3	Col 4
Row 1	0, 0	0, 1	0, 2	0, 3
Row 2	1, 0	1, 1	1, 2	1, 3
Row 3	2, 0	2, 1	2, 2	2, 3
Row 4	3, 0	3, 1	3, 2	3, 3

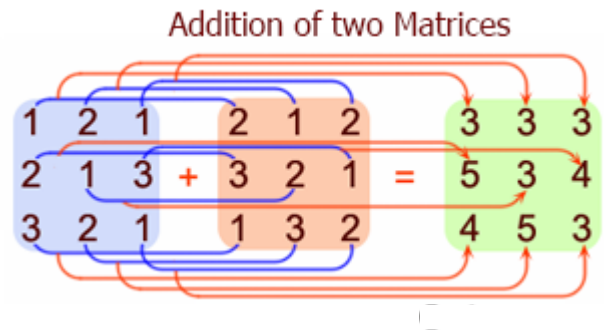
```
using System;




class arrex8
{
    private static void Main()
    {
        Console.WriteLine("enter array size: ");
        int m = Convert.ToInt32(Console.ReadLine());
        int n = Convert.ToInt32(Console.ReadLine());
        int[,] arr1 = new int[m, n];

        Console.WriteLine("output:");
        for (int i = 0; i < m; i++)
        {
            for (int j = 0; j < n; j++)
            {
                Console.Write(i + "," +j+ "   ");
            }
            Console.WriteLine();
        }
    }
}
```

Output:

```
enter array size:
4
4
output:
0,0   0,1   0,2   0,3
1,0   1,1   1,2   1,3
2,0   2,1   2,2   2,3
3,0   3,1   3,2   3,3
```

7.5.9 Array: Exercise-9 with Solution

Write a program in C# Sharp for addition of two Matrices of same size.



Sample Solution:-

C# Sharp Code:

```
using System;

internal class arrex8
{
    private static void Main()
    {
        Console.WriteLine("enter array size: ");
        int m = Convert.ToInt32(Console.ReadLine());
        int n = Convert.ToInt32(Console.ReadLine());
        int[,] arr1 = new int[m, n];
        int[,] arr2 = new int[m, n];
        int[,] arr3 = new int[m, n];
        Console.WriteLine("output:");
        for (int i = 0; i < m; i++)
        {
            for (int j = 0; j < n; j++)
            {
                arr1[i, j] = Convert.ToInt32(Console.ReadLine());
            }
        }

        for (int i = 0; i < m; i++)
        {
            for (int j = 0; j < n; j++)
            {
                arr2[i, j] = Convert.ToInt32(Console.ReadLine());
            }
        }
```

```
        for (int i = 0; i < m; i++)
        {
            for (int j = 0; j < n; j++)
            {
                arr3[i, j] = arr1[i, j] + arr2[i, j];
            }
        }

        for (int i = 0; i < m; i++)
        {
            for (int j = 0; j < n; j++)
            {
                Console.Write(arr3[i, j] + " ");
            }
            Console.WriteLine();
        }
    }
}
```

Output:

enter array size:

3

3

output:

```
1
2
3
4
5
6
7
8
9
1
1
1
2
2
2
3
3
3
2 3 4
6 7 8
10 11 12
```