**Session 7: Array :Types of Array And Foreach.**

* **Array :-** array is a collection of element. Its refrence type. Not allocate memory at the time of declaration but at run time allocate. Which contain multiple values in single variable. [ ] used to indicate array. Its similar (Single) datatype. It uses the indexer. And index value start from 0. It allocate the adjacent memory at the time of allocation. Accessing value can be done using indexer value. Default values is null. Support zero based and not include negative index. It goes upto n-1. Easy to manage & sorting. Length is used to check size of array. new keyword allocate the memory in ram. Its an object datatype.
* **Foreach :** used with array, collection, list. No need to provide condition. Iterate with each element. Used for refrence type. Used to perform large data fetching.
* **Array Initialise From User:** user can take the values and initialised this values in one container.
* **Types of Arrays :- 1] Single Dimensional Array:** elements store in single row. Declaration and initialisation are same.

**2] Multidimensional Array:**  store data in combination more than one. Array is represented as row and column. It indicate element as excel sheet. Called array of array. Getlength is used to check the dimension. Rank is used to display the size of dimension.

**Types: a) Rectangular Array :-** whatever size is fixed that contain. All dimension has fixed or constant value. Same colum in each row.

**b) Jagged Array:-** no need to specify size. Elements of array. Dimension value cannot be fixed.

**S7\_\_ArraysAndTypes.cs**

using System;

using System.Collections.Generic;

using System.Text;

namespace AllSession

{

class S7\_\_ArraysAndTypes

{

//\*\*\*\*\*\*\*\*\*\*\*Defining Array\*\*\*\*|| Single Dimensioanl Array\*\*\*\*\*\*

//dataType[] variable0 = new dataType[5];

//dataType[] variable1 = new dataType[] { "abc","abc","abc"};

//dataType[] variable2 = { 1.0, 2.0, 3.0, 4.0, 5.0 };

//dataType[] variabel1, variable2;

public void SingleDimensionalArray()

{

int[] number = new int[5];

Console.WriteLine($"Numbers Are: {number[0] = 1} , {number[1] = 11}, {number[2] = 110},{number[3] = 'C'} ,{number[4] = 'D'} \n");

//Console.WriteLine($"Char is: {number[5] = 'Z'}"); //Out of Range

//Console.WriteLine($"{number[4] = "C"}"); //Having Error

//Using ForEach

string[] city = { "Panvel", "Pune", "Nanded", "Latur", "Beed", "Mumbai", "Dubai" };

Console.Write($"Length of Array is : {city.Length} \n\nCities Are: \t");

foreach (string cities in city)

Console.Write(cities +", ");

//User Accepted Array

Console.Write("\n\nHow Many Numbers you want: ");

int userArrray = Convert.ToInt32(Console.ReadLine());

int[] arrays = new int[userArrray];

for (int i = 0; i < userArrray; i++)

{

Console.Write("Enter Data Value: ");

int finalyArray = Convert.ToInt32(Console.ReadLine());

arrays[i] = finalyArray;

}

Console.WriteLine("\n\*\*\*\*\*\* Your Final Array is: \*\*\*\*\*\*\*\*\*\*\*\*");

Console.Write("Final Array is: ");

foreach(int data in arrays)

Console.Write(data +" ");

}

////\*\*\*MultiDimensioanl Dimensioanl Array\*\*\*\*\*\*

/// Rectanuglar Array

// dataTtype[Row,Clm] variable = new dataType[Row, Clm];

// dataTtype[Row, Clm] variable = new dataType[Row, Clm]{ { {1,2,3},{4,5,6},{7,8,9} } };

// dataTtype[Row, Clm] variable = { { {1,2,3},{4,5,6},{7,8,9} } };

// dataTtype[Row, Clm] variable = new dataType[Row, Clm];

// Jagged Array

//dataTyoe[Row][Colm] variable = new dataType[Row][Colum] JaggedArray

// 0 1 2 3 <<Col

// Row>> 0 {12,24,36,48 }, [2,2] ==20

// 1 {60,72,84,96 },

// 2 {10,20,20,40 },

public void MultiDimensionalArray()

{

int[,] numbers = new int[3,4]

{

{12,24,36,48 }, // Rectangular Array.

{60,72,84,96 },

{10,20,30,40 },

};

Console.WriteLine($"\nElements of Array is: {numbers[1,2]} \n");

Console.WriteLine($"Dimension Array is: { numbers.GetLength(1)} \t Size of Dimension {numbers.Rank} " +"\n");

Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\* For Loop \*\*\*\*\*\*\*\* \n");

for (int i = 0; i < numbers.GetLength(0); i++)

{

for (int j = 0; j < numbers.GetLength(1); j++)

{

Console.Write(numbers[i, j] + " ");

}

Console.WriteLine();

}

foreach (int number in numbers)

Console.Write(number + " ");

// Jagged Array

int[][] numericalValues = new int[3][];

numericalValues[0] = new[] { 12, 24, 36, 48,99,88,77 };

numericalValues[1] = new[] { 60, 72, 84, 96,00 };

numericalValues[2] = new[] { 10, 20, 30,44 };

Console.WriteLine("\n\*\*\*\*\*\*\*\* ForeachLoop \*\*\*\*\*\*\*\*\*\*");

foreach (int[] number in numericalValues)

{

foreach (int i in number)

{

Console.Write(i + " ");

}

Console.WriteLine();

}

Console.WriteLine($"\nValue of JaggedArray is: {numericalValues[0][6]}");

}

}

}

**Program.cs**

using AllSession;

using Session1.nestedNamespace; // Assembly Refrence OR Namespace OR PAckage

using System;

namespace Session1

{

class Program : S4\_AccessModifiereAndKeyword

{

static void Main(string[] args)

{

//Session 7 Array and Types

S7\_\_ArraysAndTypes array = new S7\_\_ArraysAndTypes();

array.SingleDimensionalArray();

array.MultiDimensionalArray();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*PROGRM\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Q] Sort Array and Reverse Array.

Q] Sort Array and Reverse Array 2D. Swap The Array.