|  |
| --- |
| Day 13 Assignment (09-02-2022)  By  M.Pallavi |

|  |
| --- |
| Program 1: Declare a 2 dimensional array of size (2,2) and initialize using indexes and print the values using nested for loop. |
| Code:  using System;  namespace pgmon2DArray  {  internal class Program  {  static void Main(string[] args)  {  int[,] data = new int[2,2];  data[0, 0] = 2;  data[0, 1] = 3;  data[1, 0] = 4;  data[1, 1]= 5;  for (int i = 0; i < 2; i++)  {  for (int j = 0; j < 2; j++)  {  Console.Write(data[i,j] + "\t");  }  Console.ReadLine("\n");  }  }  }  } |
| Output: |

|  |
| --- |
| Program 2: Declare a 2-D array of size (3,2) and initialize in the same line while declaring and print the values using nested for loop |
| Code:  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace pgm2on2Darray  { /\*Declare a 2-D array of size (3,2) and initialize in the same line while declaring and print the values using nested for loop\*/  internal class Program  {  static void Main(string[] args)  {  int[,] data = new int[, ] { { 1, 2 }, { 4, 1 }, { 3, 4 } };  for (int i = 0; i < 3; i++)  {  for (int j = 0; j < 2; j++)  {  Console.Write(data[i, j] + " ");  }  Console.ReadLine();  } }  }  } |
| Output: |

|  |
| --- |
| Program 3: Declare a 2-D array of size (3,3) and print trace of the array |
| Code:  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace program3  {  internal class Program  { /\* program on 2D array to print trace of array\*/  static void Main(string[] args)  {  int[,] data = new int[,] { { 1, 2, 3 }, { 4, 1, 8 }, { 3, 4, 9 } };  int sum = 0;  for (int i = 0; i < 3; i++)  {  for (int j = 0; j < 3; j++)  {  Console.Write(data[i, j] + " ");  }  Console.ReadLine();  }  for (int i = 0; i < 3; i++)  {  for (int j = 0; j < 3; j++)  {  if (i == j)  {  sum = sum + data[i, j];  }  }  }  Console.WriteLine("\n The Trace of the array is : {0}", sum);  Console.ReadLine();  }  }  } |
| Output: |

|  |
| --- |
| Program 4: Declare a 2-D array of size (2,2) and read values from user and print the array values. |
| Code:  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace program4  {  internal class Program  {  static void Main(string[] args)  {  int[,] data = new int[2, 2];    for (int i = 0; i < 2; i++)// Reading elements from the user  {  for (int j = 0; j < 2; j++)  {  Console.Write("Enter the array elements ");  data[i, j] = Convert.ToInt32(Console.ReadLine());  }  }  // Printing the Array elements  Console.WriteLine(" array items are");  for (int i = 0; i < 2; i++)  {  for (int j = 0; j < 2; j++)  {  Console.Write("\t" + data[i, j] + " ");  }  Console.ReadLine();  }    }  }  } |
| Output: |

|  |
| --- |
| 5. Declare TWO 2-D arrays of size ( 2, 2) and read values from user and print the sum of the two matrices. |
| Code:  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace program5  { /\* Author : M.Pallavi  Purpose : program to print sum of two matrices.  \*/  internal class Program  {  static void Main(string[] args)  {  int[,] array1 = new int[2, 2];  int[,] array2= new int[2, 2];  int[,] array3 = new int[2, 2];  Console.WriteLine("enter first matrix");  //read 1st matrix  for (int i = 0; i < 2; i++)  {  for (int j = 0; j < 2; j++)  {  array1[i, j] = Convert.ToInt32(Console.ReadLine());  }  }  //read 2nd matrix  Console.WriteLine("enter second matrix");  for (int i = 0; i < 2; i++)  {  for (int j = 0; j < 2; j++)  {  array2[i, j] = Convert.ToInt32(Console.ReadLine());  }  }  for (int i = 0; i < 2; i++)  {  for (int j = 0; j < 2; j++)  {  array3[i, j] = array1[i, j] + array2[i, j];  }  }  Console.WriteLine($"sum of 2 matrices is ");  for ( int i = 0; i < 2; i++)  {  Console.WriteLine("\n");  for (int j=0; j < 2; j++)  {    Console.Write(("\t" + array3[i, j] + " ") );  }  }  Console.WriteLine("\n");  Console.ReadLine();  }    }  } |
| Output: |

|  |
| --- |
| 6. Declare TWO 2-D arrays of size (2,2) and read values from user and print the product of the two matrices**.** |
| **Code:**  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace program6  {  internal class Program  {  static void Main(string[] args)  {  int[,] array1 = new int[2, 2];  int[,] array2 = new int[2, 2];  int[,] array3 = new int[2, 2];  Console.WriteLine("enter first matrix");  //read 1st matrix  for (int i = 0; i < 2; i++)  {  for (int j = 0; j < 2; j++)  {  array1[i, j] = Convert.ToInt32(Console.ReadLine());  }  }  //read 2nd matrix  Console.WriteLine("enter second matrix");  for (int i = 0; i < 2; i++)  {  for (int j = 0; j < 2; j++)  {  array2[i, j] = Convert.ToInt32(Console.ReadLine());  }  }  for ( int i = 0; i < 2; i++)  {  for ( int j = 0; j < 2; j++)  {    for (int k = 0; k < 2; k++)  {  array3[i, j] += array1[i, k] \* array2[k, j];  }  }  }  Console.WriteLine("The product of the two matrices is :");  for (int i = 0; i < 2; i++)  {  Console.WriteLine("\n");  for (int j = 0; j < 2; j++)  {  Console.Write(("\t" + array3[i, j] + " "));  }  }  Console.WriteLine("\n");  Console.ReadLine();  }  }  } |
| **Output:** |

|  |
| --- |
| Program 6 :write a c# program to declare a jagged array and print values |
| **Code:**  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace pgmOnJaggedArray  {  internal class Program  {  static void Main(string[] args)  {  char[][] names = new char[6][];  names[0] = new char[] { 'r', 'a', 'm','e','s','h' };  names[1] = new char[] { 'k', 'r', 'i', 't', 'i' };  names[2] = new char[] { 'n', 'b', 't', 'e', 'c', 'h', 'n', 'o', 'l', 'o', 'g', 'e', 's' };  names[3] = new char[] { 'p', 'a', 'l', 'l', 'a', 'v', 'i' };  names[4] = new char[] { 'c', 'h', 'a', 'i', 't', 'h', 'u' };  names[5] = new char[] { 'c', 'o', 'm', 'p', 'a', 'n', 'y' };  Console.WriteLine("printing array values ");  for (int i = 0; i < 6; i++)  {  for (int j = 0; j < names[i].Length; j++)  {  Console.Write(names[i][j]);  }  Console.ReadLine();  }  }  }  } |
| **Output:** |

**7. What is a jagged array ? What is the benefit of jagged array**

* A jagged array is an array whose elements are arrays.
* The elements of a jagged array can be of different dimensions and sizes
* . A jagged array is sometimes called an "array of arrays."
* A Jagged Array is an **array of an array in** which the length of each array index can differ.

USES:

* jagged Array Use jagged arrays to**store data that comes in different sizes, and to reduce memory usage.**

9. What is Recursion?

The process in which a function calls itself directly or indirectly is called **recursion.**

|  |
| --- |
| **Program: Write a C# program to illustrate usage of Stack<> .Write couple of points about Stack** |
| **Code:**  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace pgmOnStack  {  internal class Program  {  static void Main(string[] args)  {  Stack<int> data = new Stack<int>();  data.Push(1);  data.Push(2);  data.Push(3);  data.Push(4);  data.Push(5);  Console.WriteLine("total no.elements are " +data.Count);  Console.WriteLine("poped element is " +data.Pop());  Console.WriteLine("peek element is"+data.Peek());  Console.WriteLine("count is " +data.Count);  Console.ReadLine();  }  }  } |
| **Output:** |

Stack is a special type of collection that stores elements in LIFO style (Last In First Out). C# includes the generic Stack<T>

* Elements can be added using the Push() method. Cannot use collection-initializer syntax.
* Elements can be retrieved using the Pop() and the Peek() methods. It does not support an indexer

|  |
| --- |
| 11.Program: Write a C# program to illustrate usage of Queue. |
| Code:  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace pgmQueue  {  internal class Program  {  static void Main(string[] args)  {  Queue<int> data = new Queue<int>();  data.Enqueue(1);  data.Enqueue(2);  data.Enqueue(3);  data.Enqueue(4);  data.Enqueue(5);  data.Enqueue(6);  Console.WriteLine("no of elements in queue is " + data.Count);  Console.WriteLine("the element removed is " + data.Dequeue());  Console.WriteLine("last element is " + data.Peek());  Console.WriteLine("no of elements is " + data.Count);  Console.ReadLine();  }  }  } |
| Output: |

**B.**Write couple of points about Queue?

* Queue<T> is FIFO (First In First Out) collection.
* It comes under System. Collection. Generic namespace.
* Queue<T> can contain elements of the specified type
* Elements can be added using the Enqueue() method.
* Elements can be retrieved using the Dequeue() and the Peek() methods