Feb 9th Assignment

By

Surya Teja Chandolu

|  |
| --- |
| 1. Declare a 2 dimentional array of size (2,2) and initialize using indexes and print the values using nested for loop |
| Code: |
| using System;  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Author: Surya Teja  \* Purpose: Declare a 2 dimentional array of size (2,2) and initialize using indexes and print the values  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  namespace Array1  {  internal class Program  {  static void Main(string[] args)  {  int[,] data = new int[,]{ { 1, 2 }, { 3, 4 } };  for (int i = 0; i < data.GetLength(0); i++)  {  for (int j = 0; j < data.GetLength(1); j++)  {  Console.Write($"{data[i, j]} ");  }  Console.WriteLine();  }  Console.ReadLine();  }  }  } |
| Output: |
|  |

|  |
| --- |
| 1. 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;  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Author: Surya Teja  \* Purpose: 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  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  namespace Array1  {  internal class Program  {  static void Main(string[] args)  {  int[,] data = new int[,]{ { 1, 2 }, { 3, 4 }, { 5, 6 } };  for (int i = 0; i < data.GetLength(0); i++)  {  for (int j = 0; j < data.GetLength(1); j++)  {  Console.Write($"{data[i, j]} ");  }  Console.WriteLine();  }  Console.ReadLine();  }  }  } |
| Output: |
|  |

|  |
| --- |
| 1. Declare a 2-D array of size (3,3) and print trace of the array |
| Code: |
| using System;  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Author: Surya Teja  \* Purpose: Declare a 2-D array of size (3,3) and print trace of the array  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  namespace Array1  {  internal class Program  {  static void Main(string[] args)  {  int sum = 0;  int[,] data = new int[,]{ { 1, 2, 3 }, { 3, 4, 5 }, { 5, 6, 7 } };  for (int i = 0; i < data.GetLength(0); i++)  {  for (int j = 0; j < data.GetLength(1); j++)  {  if(i == j)  sum = sum + data[i, j];  }  }  Console.WriteLine($"Sum of trace is {sum}");  Console.ReadLine();  }  }  } |
| Output: |
|  |

|  |
| --- |
| 1. Declare a 2-D array of size (2,2) and read values from user and print the array values. |
| Code: |
| using System;  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Author: Surya Teja  \* Purpose: Declare a 2-D array of size (2,2) and read values from user and print the array values.  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  namespace Array1  {  internal class Program  {  static void Main(string[] args)  {  int sum = 0;  int[,] data = new int[2, 2];  //User input  for (int i = 0; i < data.GetLength(0); i++)  {  for (int j = 0; j < data.GetLength(1); j++)  {  Console.Write($"Enter array number at [{i}, {j}]: ");  data[i,j] = Convert.ToInt32(Console.ReadLine());  }  }  //Print  for (int i = 0; i < data.GetLength(0); i++)  {  for (int j = 0; j < data.GetLength(1); j++)  {  Console.Write($"{data[i, j]} ");  }  Console.WriteLine();  }  Console.ReadLine();  }  }  } |
| Output: |
|  |

|  |
| --- |
| 1. 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;  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Author: Surya Teja  \* Purpose: Declare TWO 2-D arrays of size (2,2) and read values from user and print the sum of the two matrices.  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  namespace SumOfTwoMatrices  {  internal class Program  {  static void Main(string[] args)  {  int[,] data1 = new int[2, 2];  int[,] data2 = new int[2, 2];  int[,] data3 = new int[2, 2];  //User input for data1  for (int i = 0; i < data1.GetLength(0); i++)  {  for (int j = 0; j < data1.GetLength(1); j++)  {  Console.Write($"Enter array number at [{i}, {j}]: ");  data1[i, j] = Convert.ToInt32(Console.ReadLine());  }  }  //Print for data1  for (int i = 0; i < data1.GetLength(0); i++)  {  for (int j = 0; j < data1.GetLength(1); j++)  {  Console.Write($"{data1[i, j]} ");  }  Console.WriteLine();  }  //User input for data2  for (int i = 0; i < data2.GetLength(0); i++)  {  for (int j = 0; j < data2.GetLength(1); j++)  {  Console.Write($"Enter array number at [{i}, {j}]: ");  data2[i, j] = Convert.ToInt32(Console.ReadLine());  }  }  //Print for data2  for (int i = 0; i < data2.GetLength(0); i++)  {  for (int j = 0; j < data2.GetLength(1); j++)  {  Console.Write($"{data2[i, j]} ");  }  Console.WriteLine();  }  //Logic for sum of two matrices  for (int i = 0; i < data3.GetLength(0); i++)  {  for (int j = 0; j < data3.GetLength(1); j++)  {  data3[i, j] = data1[i, j] + data2[i, j];  }  }  //Print sum of two matrices  Console.WriteLine("Sum of two Matrices are: ");  for (int i = 0; i < data3.GetLength(0); i++)  {  for (int j = 0; j < data3.GetLength(1); j++)  {  Console.Write($"{data3[i, j]} ");  }  Console.WriteLine();  }  Console.ReadLine();  }  }  } |
| Output: |
|  |

|  |
| --- |
| 1. 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;  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Author: Surya Teja  \* Purpose: Product Of Matrix  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  namespace ProductOfMatrix  {  internal class Program  {  static void Main(string[] args)  {  int f1, f2, s1, s2;  //Read Data  Console.Write("Enter Row size of First matrix: ");  f1 = Convert.ToInt32(Console.ReadLine());  Console.Write("Enter Colomn size of First matrix: ");  f2 = Convert.ToInt32(Console.ReadLine());  Console.Write("Enter Row size of Second matrix: ");  s1 = Convert.ToInt32(Console.ReadLine());  Console.Write("Enter Colomn size of Second matrix: ");  s2 = Convert.ToInt32(Console.ReadLine());  //Array  int[,] first = new int[f1, f2];  int[,] second = new int[s1, s2];  int[,] product = new int[f1, s2];    //User input for first matrix  for (int i = 0; i < f1; i++)  {  for (int j = 0; j < f2; j++)  {  Console.Write($"Enter array number at [{i}, {j}]: ");  first[i, j] = Convert.ToInt32(Console.ReadLine());  }  }  //Print for first matrix  for (int i = 0; i < f1; i++)  {  for (int j = 0; j < f2; j++)  {  Console.Write($"{first[i, j]} ");  }  Console.WriteLine();  }  //User input for second matrix  for (int i = 0; i < s1; i++)  {  for (int j = 0; j < s2; j++)  {  Console.Write($"Enter array number at [{i}, {j}]: ");  second[i, j] = Convert.ToInt32(Console.ReadLine());  }  }  //Print for second matrix  for (int i = 0; i < s1; i++)  {  for (int j = 0; j < s2; j++)  {  Console.Write($"{second[i, j]} ");  }  Console.WriteLine();  }  //Logic for mutlipcation table  if(f2 == s1)  {  for(int i = 0; i < f1; i++)  {  for(int j = 0; j < s2; j++)  {  product[i, j] = 0;  for(int k = 0; k < f2; k++)  {  product[i, j] += first[i, k] \* second[k, j];  }  }  }  //Print Mul Table  for(int i = 0; i < f1; i++)  {  for(int j = 0; j < s2; j++)  {  Console.Write($"{product[i, j]} ");  }  Console.WriteLine();  }  }  Console.ReadLine();  }  }  } |
| Output: |
|  |

|  |
| --- |
| 1. What is a jagged array What is the benefit of jagged array |
| * A jagged array is an array whose elements are arrays, possibly of different sizes. A jagged array is sometimes called an "array of arrays." * Each of the elements is a single-dimensional array of integers. * It makes things easy where there is a need to store data in a multidimensional way using the same variable name. * It helps in memory management which makes the program to be executed very smoothly and fast as well. |

|  |
| --- |
| 1. WACP 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;  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Author: Surya Teja  \* Purpose: Jagged Array  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  namespace JaggedArray  {  internal class Program  {  static void Main(string[] args)  {  char[][] names = new char[3][];  names[0] = new char[] { 'N','I','K','E' };  names[1] = new char[] { 'A','D','I','D','A','S' };  names[2] = new char[] { 'W','O','O','D','L','A','N','D' };  // Display the array elements.  for (int i = 0; i < names.Length; i++)  {  Console.Write($"Name at position {i} is ");  for (int j = 0; j < names[i].Length; j++)  {  Console.Write(names[i][j]);  }  Console.WriteLine();  }  Console.ReadKey();  }  }  } |
| Output: |
|  |

|  |
| --- |
| 1. What is Recursion |
| * A recursion function is which call itself again and again until the condition satisfy. * Thes will call function with parameters and receive new parameter after every execution. |

|  |
| --- |
| 1. WACP to illustrate usage of Recursion. What are the benefits of recursion |
| Code: |
| using System;  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Author: Surya Teja  \* Purpose: Factorial Recursion  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  namespace FactorialRecursion  {  internal class Program  {  class Factorial  {  public int Fact(int number)  {  if (number == 1)  return 1;  else  return number \* Fact(number - 1);  }  }  static void Main(string[] args)  {  int input;  Console.Write("Enter number: ");  input = Convert.ToInt32(Console.ReadLine());  Factorial f = new Factorial();  Console.WriteLine($"Fact is {f.Fact(input)}");  Console.ReadLine();  }  }  } |
| Output: |
|  |

|  |
| --- |
| 1. WACP to illustrate usage of Stack<> Write couple of points about Stack |
| Code: |
| using System;  using System.Collections.Generic;  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Author: Surya Teja  \* Purpose: Stack  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  namespace StackB  {  internal class Program  {  static void Main(string[] args)  {  Stack<int> data = new Stack<int>();  data.Push(10);  data.Push(20);  data.Push(30);  for (int i = 0; i <= data.Count; i++)  Console.WriteLine($"Element Pop {data.Pop()}");  Console.WriteLine($"Element Peek {data.Peek()}");  Console.ReadLine();  }  }  } |
| Output: |
|  |

|  |
| --- |
| 1. WACP to illustrate usage of Queue<> Write couple of points about Stack |
| Code: |
| using System;  using System.Collections.Generic;  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* Author: Surya Teja  \* Purpose: Queue  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  namespace QueueB  {  internal class Program  {  static void Main(string[] args)  {  Queue<int> data = new Queue<int>();  data.Enqueue(10);  data.Enqueue(20);  data.Enqueue(30);  for (int i = 0; i <= data.Count; i++)  Console.WriteLine($"Element Remove {data.Dequeue()}");  Console.WriteLine($"Element Peek {data.Peek()}");  Console.ReadLine();  }  }  } |
| Output: |
|  |