|  |
| --- |
| Day 13(9 feb) Assignment  By P Ramakrishna |

|  |
| --- |
| 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;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace \_2d\_array  {  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* author: Ramakrishna  \* purpose: Declare a 2 dimentional array of size (2,2)  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  internal class Program  {  static void Main(string[] args)  {  int[,] data = new int[2, 2];  data[0, 0] = 1;  data[0, 1] = 6;  data[1,0] = 45;  data[1,1] = 12;  for ( int i=0;i<2;i++)  {  for (int j = 0; j<2; j++)  {  Console.Write(data[i, j]+" ");  }  Console.Write("\n");  }  Console.ReadLine();  }  }  } |
| Output: |

|  |
| --- |
| 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 day\_13\_\_project\_2  {  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* author: Ramakrishna  \* purpose: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[,] { { 5, 6 }, { 8, 4 }, { 11, 22 } };  for (int i = 0; i<3; i++)  {  for (int j = 0; j<2; j++)  {  Console.Write(data[i, j]+" ");  }  Console.Write("\n");  }  Console.ReadLine();  }  }  } |
| Output: |

|  |
| --- |
| 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 day\_13\_\_project\_3  {  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* author: Ramakrishna  \* purpose: Print trace of array  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  internal class Program  {  static void Main(string[] args)  {  int sum =0;  int[,] data = new int[,] { { 5, 6,7 }, { 8, 4,3 }, { 11, 22,33 } };  for (int i = 0; i<3; i++)  {  for (int j = 0; j<3; j++)  {  if(i==j)  sum = sum + data[i,j];    }    }  Console.WriteLine(sum);  Console.ReadLine();  }  }  } |
| Output: |

|  |
| --- |
| 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 day\_13\_project\_4  {  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* author: Ramakrishna  \* purpose:Declare a 2-D array and read values from user  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  internal class Program  {  static void Main(string[] args)  {  int[,] data = new int[2, 2];  for (int i = 0; i < 2; i++)  {  for (int j = 0; j < 2; j++)  {  Console.WriteLine("enter array value");  data[i,j]=Convert.ToInt32(Console.ReadLine());  }  }  for (int i = 0; i<2; i++)  {  for (int j = 0; j<2; j++)  {  Console.Write($"{data[i,j]} ");  }  Console.Write("\n");  }    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 Day13\_project6  {  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* author: Ramakrishna  \* purpose:sum of two matrices  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  internal class Program  {  static void Main(string[] args)  {  int[,] data = new int[2, 2];  for (int i = 0; i < 2; i++)  {  for (int j = 0; j < 2; j++)  {  Console.WriteLine("enter 1st array value");  data[i, j]=Convert.ToInt32(Console.ReadLine());  }  }  for (int i = 0; i<2; i++)  {  for (int j = 0; j<2; j++) { }  }  int[,] data2 = new int[2, 2];  for (int i = 0; i < 2; i++)  {  for (int j = 0; j < 2; j++)  {  Console.WriteLine("enter 2nd array value");  data2[i, j]=Convert.ToInt32(Console.ReadLine());  }  }  for (int i = 0; i<2; i++)  {  for (int j = 0; j<2; j++)  {  Console.Write(data[i, j]+data2[i, j]+" ");  }  Console.Write("\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 Day13\_project7  {  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* author: Ramakrishna  \* purpose:product of two matrices  \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  internal class Program  {  static void Main(string[] args)  {  int[,] data = new int[2, 2];  for (int i = 0; i < 2; i++)  {  for (int j = 0; j < 2; j++)  {  Console.WriteLine("enter 1st array value");  data[i, j]=Convert.ToInt32(Console.ReadLine());  }  }  for (int i = 0; i<2; i++)  {  for (int j = 0; j<2; j++) { }  }  int[,] data2 = new int[2, 2];  for (int i = 0; i < 2; i++)  {  for (int j = 0; j < 2; j++)  {  Console.WriteLine("enter 2nd array value");  data2[i, j]=Convert.ToInt32(Console.ReadLine());  }  }  for (int i = 0; i<2; i++)  {  for (int j = 0; j<2; j++)  {  Console.Write(data[i, j]\*data2[i, j]+" ");  }  Console.Write("\n");  }  Console.ReadLine();  }  }  } |
| Output: |

|  |
| --- |
| 7. What is a jagged array What is the benefit of jagged array |
| Jagged array is a **array of arrays** such that member arrays can be of different sizes. In other words, the length of each array index can differ. The elements of Jagged Array are reference types and initialized to null by default. Jagged Array can also be mixed with multidimensional arrays. Here, the number of rows will be fixed at the declaration time, but you can vary the number of columns. |
| **Benfits of jagged array:**  There are several benefits of using jagged arrays. One of the most crucial advantages is that it makes things easy where there is a need to store data in a multidimensional way using the same variable name. Furthermore, it helps in memory management which makes the program to be executed very smoothly and fast as well. |

|  |
| --- |
| 8. 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;  namespace Day\_13\_Project\_8  {  internal class Program  {  static void Main(string[] args)  {  char[][] names = new char[3][];  names[0]= new char[] { 'R', 'K' };  names[1] = new char[] { 'k', 'r', 'i', 's', 'h', 'n', 'a' };  names[2] = new char[] { 'S', 'a', 'n', 'j', 'a', 'y' };  for(int i=0;i<3;i++)  {  for(int j=0;j<names[i].Length;j++)  {  Console.Write(names[i][j]);  }  Console.Write("\n");  Console.ReadLine();  }  }  }  } |
| Output: |

|  |
| --- |
| 9. What is Recursion |
| A recursive method is a method which calls itself again and again on basis of few statements which need to be true. |
| Benfits of Recursion:  Recursion helps in reducing the length of the code.  It provides a clean and straightforward way to write the code. |

|  |
| --- |
| 10 . WACP to illustrate usage of Recursion.  What are the benefits of recursion |
| Code: using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Day\_13\_project\_9  {  internal class Program  {      static void Main(string[] args)  {  int fact = 1;  Console.WriteLine("Enter a number:");  int num = int.Parse(Console.ReadLine());  for (int i = 1; i <=num; i++)  {  fact = fact\*i;  }  Console.WriteLine($"fact of given number is:{fact}");  Console.ReadLine();    }  }  } |
| Output: |

|  |
| --- |
| Write couple of points about Stack |
| * A stack is a last in ,first –out collection of objects.it is used when you need last in,first out access to terms.it is both genic and non generic tyoe of collection * When you add an item in the list, it is called push. * when you remove an item, it is called pop. |
| Write couple of points Queue |
| * A queue is a first in first out collection of objects. * When you add an item in the list, it is called enqueue. * when you remove an item, it is called dequeue. |

|  |
| --- |
| 11.WACP to illustrate usage of Stack<> |
| Code : using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace day\_11\_project\_10  {  internal class Program  {  static void Main(string[] args)  {  Stack<int> data = new Stack<int>();  data.Push(20);  data.Push(4);  data.Push(55);  data.Push(12);  Console.WriteLine(data.Count);  Console.WriteLine(data.Pop());  Console.WriteLine(data.Count);  Console.ReadLine();  }  }  } |
| Output:  ,peek |

|  |
| --- |
| 12. WACP to illustrate usage of Queue<> |
| Code: using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Day\_13\_Project\_11  {  internal class Program  {  static void Main(string[] args)  {  Queue<int> data = new Queue<int>();  data.Enqueue(24);  data.Enqueue(35);  data.Enqueue(45);  Console.WriteLine(data.Count);  Console.WriteLine(data.Dequeue());  Console.WriteLine(data.Count);  Console.ReadLine();  }  }  } |
| Output:  ,peek |