Day 13 Assignments

By

Praveen Chakravarthi

09-02-2022

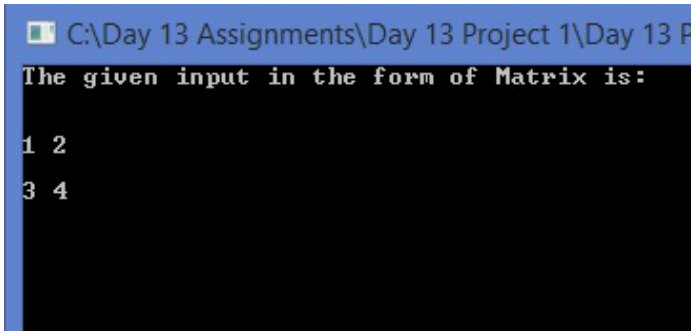
NB Health Care

1. Declare a 2 dimensional array of size (2,2) and initialise 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 Day_13_Project_1
{
    // Author : Praveen Chakravarthi
    // Purpose : 2-d Array (2,2)
    internal class Program
    {
        static void Main(string[] args)
        {
            int[,] data = new int [2,2];

            // Array initialisation using index

            data[0,0] = 1;
            data[0,1] = 2;
            data[1,0] = 3;
            data[1,1] = 4;

            Console.WriteLine("The given input in the form of Matrix is: ");
            Console.WriteLine("\n");

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

                }
                Console.WriteLine("\n");
            }
            Console.ReadLine();
        }
    }
}
```

Output:



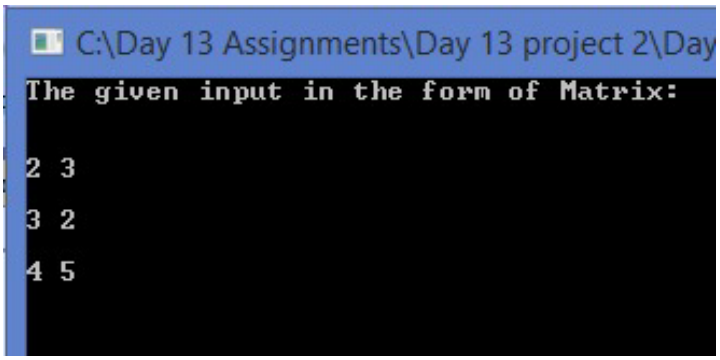
2. Declare a 2-D array of size (3,2) and initialise in the same line while declaring and print the values using nested for loop

Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Day_13_project_2
{
   // Author : Praveen Chakravarthi
   // Purpose : 2-d Array (3,2)
   internal class Program
   {
      static void Main(string[] args)
      {
         // Initialising values in same the line

         int[,] data = new int[,] { { 2, 3 },{ 3, 2 },{ 4, 5 } };

         Console.WriteLine("The given input in the form of Matrix: ");
         Console.WriteLine("\n");

         for (int i=0; i<3; i++)
         {
            for (int j=0; j<2;j++)
            {
               Console.Write(data[i,j] + " ");
            }
            Console.WriteLine("\n");
         }
         Console.ReadLine();
      }
   }
```

}

Output:



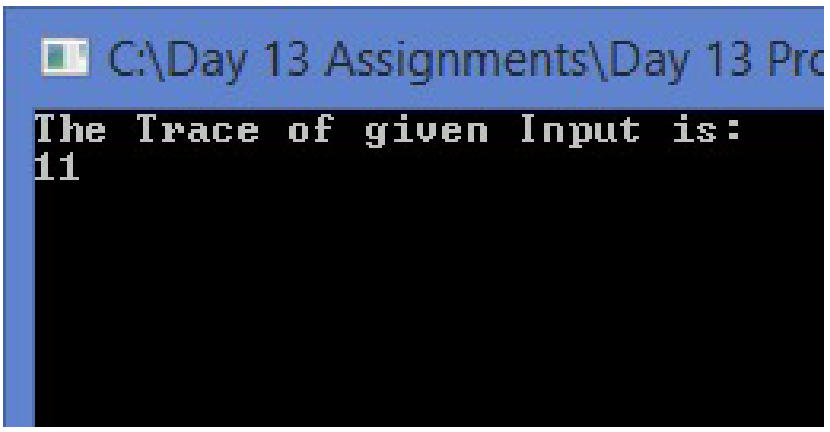
3. Declare a 2-D array of size (3,3) and print trace of the array

Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Day_13_Project_3
{
    // Author : Praveen Chakravarthi
    // Purpose : Trace of Array
    internal class Program
    {
        static void Main(string[] args)
        {
            int[,] data = new int[3, 3];
            int sum=0;

            data[0, 0] = 1;
            data[0, 1] = 3;
            data[0, 2] = 2;
            data[1, 0] = 9;
            data[1, 1] = 4;
            data[1, 2] = 5;
            data[2, 0] = 8;
            data[2, 1] = 7;
            data[2, 2] = 6;

            for(int i=0; i<3; i++)
            {
```

```
            for (int j=0; j<3; j++)
            {
                // Condition for Trace
                if (i == j)
                    sum = sum + data[i, j];
            }
        }
        Console.WriteLine("The Trace of given Input is: ");
        Console.WriteLine(sum);
        Console.ReadLine();
    }
  }
}
```

Output:



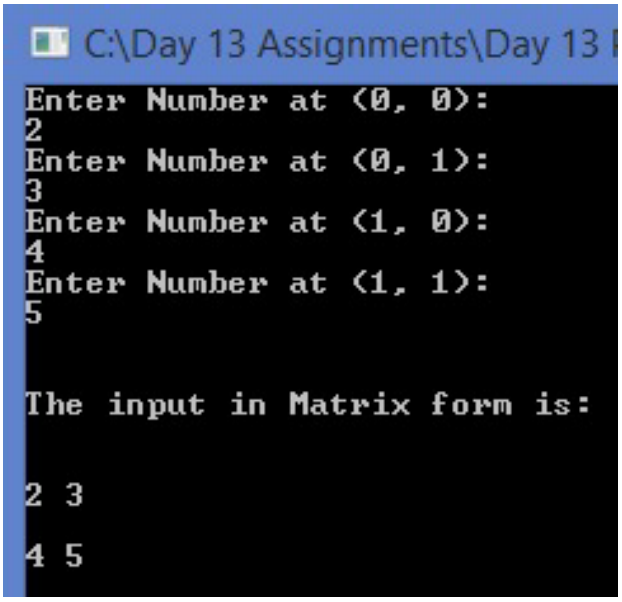
4. Declare a 2-D array of size (2,2) and read values from user and print the array values.

Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Day_13_Project_4
{
    // Author : Praveen Chakravarthi
    // Purpose : Reading Array Values from User
    internal class Program
    {
        static void Main(string[] args)
        {
```

```
            int[,] data = new int[2, 2];

            // Reading data from User
            for (int i = 0; i < 2; i++)
            {
                for (int j = 0; j < 2; j++)
                {
                    Console.WriteLine("Enter Number at {0}: ",(i,j));
                    data[i,j]=Convert.ToInt32(Console.ReadLine());
                }
            }
            Console.WriteLine("\n");

            // Printing the data given by the user
            Console.WriteLine("The input in Matrix form is: ");
            Console.WriteLine("\n");

            for (int i = 0; i < 2; i++)
            {
                for (int j = 0; j < 2; j++)
                {
                    Console.Write(data[i, j] + " ");
                }
                Console.WriteLine("\n");
            }
            Console.ReadLine();
        }
    }
}
```

Output:



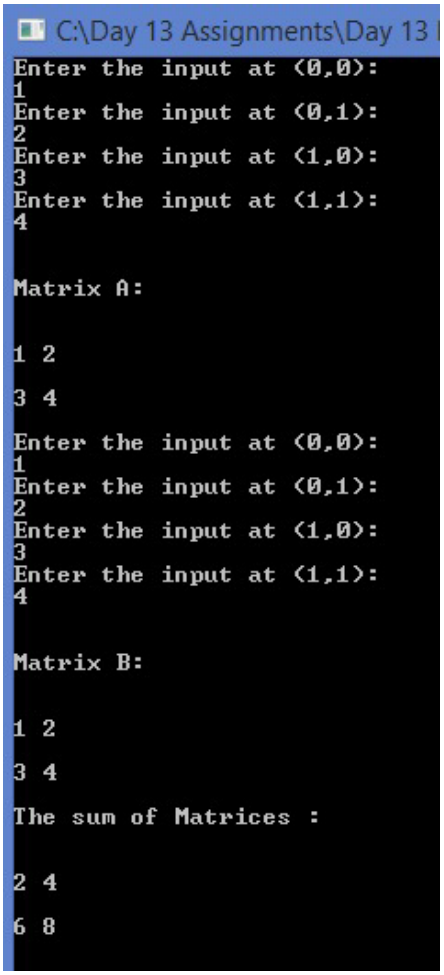
5. Declare Two 2-D arrays of size (2,2) and read values from user and print the sum of the two matrices.

Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Day_13_Project_5
{
    // Author : Praveen Chakravarthi
    // Purpose : Sum of two 2-d Arrays
    internal class Program
    {
        static void Main(string[] args)
        {
            int[,] data1 = new int[2,2];
            for (int i = 0;i<2; i++)
            {
                for (int j = 0;j<2;j++)
                {
                    Console.WriteLine($"Enter the input at ({i},{j}): ");
                    data1[i, j] = Convert.ToInt32(Console.ReadLine());

                }
            }
            Console.WriteLine("\n");
            Console.WriteLine("Matrix A: ");
            Console.WriteLine("\n");

            for (int i = 0; i < 2; i++)
            {
                for (int j = 0; j < 2; j++)
                {
                    Console.Write(data1[i, j] + " ");

                }
                Console.WriteLine("\n");
            }

            int[,] data2 = new int[2, 2];

            for (int i = 0; i < 2; i++)
            {
                for (int j = 0; j < 2; j++)
                {
                    Console.WriteLine($"Enter the input at ({i},{j}): ");
                    data2[i, j] = Convert.ToInt32(Console.ReadLine());

                }
```

```
        }
        Console.WriteLine("\n");
        Console.WriteLine("Matrix B: ");
        Console.WriteLine("\n");
        for (int i = 0; i < 2; i++)
        {
            for (int j = 0; j < 2; j++)
            {
                Console.Write(data2[i, j] + " ");

            }
            Console.WriteLine("\n");
        }

        int[,] sum = new int[2, 2];
        for (int i=0; i < 2; i++)
        {
            for (int j=0;j<2;j++)
            {
                // Adding the Matrices
                sum[i, j] = data1[i, j] + data2[i, j];
            }
        }
        Console.WriteLine("The sum of Matrices : ");
        Console.WriteLine("\n");
        for (int i = 0;i < 2; i++)
        {
            for (int j=0;j<2;j++)
            {
                Console.Write(sum[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.

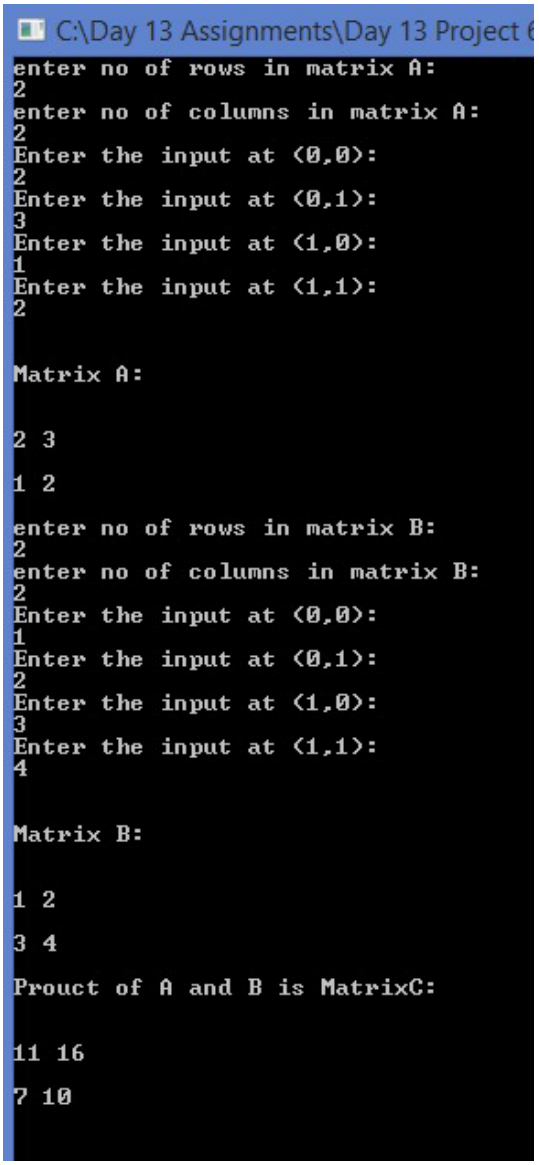
Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Day_13_Project_6
{
    // Author : Praveen Chakravarthu
    // Purpose : Product of Two 2-d Arrays

    internal class Program
    {
```

```
static void Main(string[] args)
{
    int m ;
    int n ;

    Console.WriteLine("enter no of rows in matrix A: ");
    m = Convert.ToInt32(Console.ReadLine());
    Console.WriteLine("enter no of columns in matrix A: ");
    n = Convert.ToInt32(Console.ReadLine());



    int[,] MatrixA = new int[m, n];
    for (int i = 0; i < m; i++)
    {
        for (int j = 0; j < n; j++)
        {
            Console.WriteLine($"Enter the input at ({i},{j}): ");
            MatrixA[i,j] = Convert.ToInt32(Console.ReadLine());

        }
    }
    Console.WriteLine("\n");
    Console.WriteLine("Matrix A: ");
    Console.WriteLine("\n");

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

        }
        Console.WriteLine("\n");
    }

    Console.WriteLine("enter no of rows in matrix B: ");
    int a = Convert.ToInt32(Console.ReadLine());
    Console.WriteLine("enter no of columns in matrix B: ");
    int b = Convert.ToInt32(Console.ReadLine());

    int[,] MatrixB = new int[a, b];

    for (int i = 0; i < a; i++)
    {
        for (int j = 0; j < b; j++)
        {
            Console.WriteLine($"Enter the input at ({i},{j}): ");
            MatrixB[i, j] = Convert.ToInt32(Console.ReadLine());

        }
    }
    Console.WriteLine("\n");
    Console.WriteLine("Matrix B: ");
```

```
        Console.WriteLine("\n");
        for (int i = 0; i < a; i++)
        {
            for (int j = 0; j < b; j++)
            {
                Console.Write(MatrixB[i, j] + " ");

            }
            Console.WriteLine("\n");
        }

        Console.WriteLine("Prouct of A and B is MatrixC: ");
        Console.WriteLine("\n");
        if (n == a)
        {
            int[,] MatrixC = new int[m, b];

            for (int i = 0; i < m; i++)//rows of Matrix C
            {
                for (int j = 0; j < b; j++)// columns of Matrix C
                {
                    MatrixC[i,j] = 0;

                    for (int k = 0; k < a; k++)
                    {
                        MatrixC[i,j] += MatrixA[i,k] * MatrixB[k,j];

                    }
                    Console.Write(MatrixC[i, j] + " ");
                }
                Console.WriteLine("\n");
            }
        }
        Console.ReadLine();
    }
  }
}
```

Output:



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

Jagged Array is an Array in which the length can differ in size

Benefits:

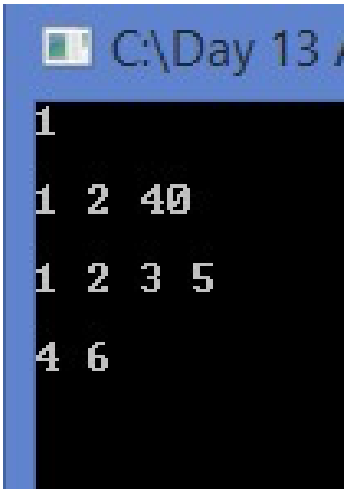
- It helps in memory management which helps the program to be executed very smoothly and fast.
- It helps in storing the values in a convinient manner.

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_Program_7
{
    // Author : Praveen Chakravarthi
    // Purpose : Jagged Arrays
    internal class Program
    {
        static void Main(string[] args)
        {
            int[][] data = new int[4][];
            data[0] = new int[] { 1 };
            data[1] = new int[] { 1, 2, 40 };
            data[2] = new int[] { 1, 2, 3, 5 };
            data[3] = new int[] { 4, 6 };

            for (int i=0; i<4; i++)
            {
                for (int j =0;j<data[i].Length; j++)
                {
                    Console.Write(data[i][j] + " ");
                }
                Console.WriteLine("\n");
            }
            Console.ReadLine();
        }
    }
}
```

Output:



9. What is Recursion? What are the benefits of Recursion?

Recursion :

- Recursion refers to the function calling itself till the specified condition is satisfied
- Recursion is used to limit the usage of variables in a code

Benefits :

- Reduces time complexity
- Makes the codes easy to write
- Reduce the length of code

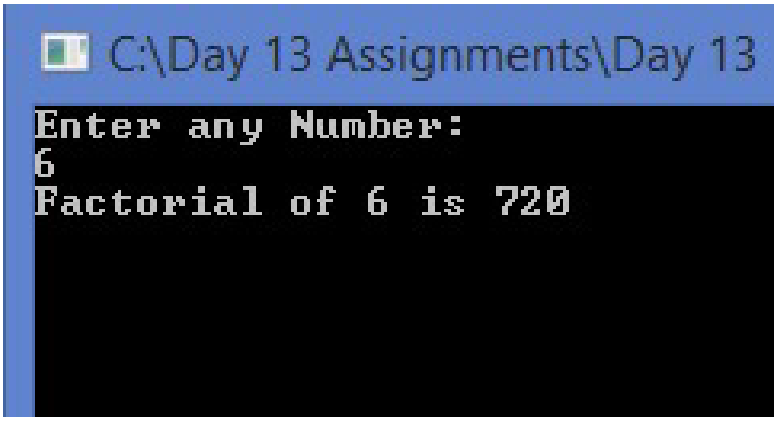
10. WACP to illustrate usage of Recursion.

Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Day_13_Project_8
{
    // Author : Praveen Chakravarthi
    // Purpose : Factorial using Recursion
```

```
    internal class Program
    {
        static int Factorial(int n)
        {
            if (n == 0)
            {
            return 1;
            }
            else
            {
                int fact = n * Factorial(n - 1);
                return fact;
            }
        }

        static void Main(string[] args)
        {
            Console.WriteLine("Enter any Number: ");
            int n = Convert.ToInt32(Console.ReadLine());

            Console.WriteLine("Factorial of {0} is {1}", n, Factorial(n));
            Console.ReadLine();
        }
    }
}
```

Output:



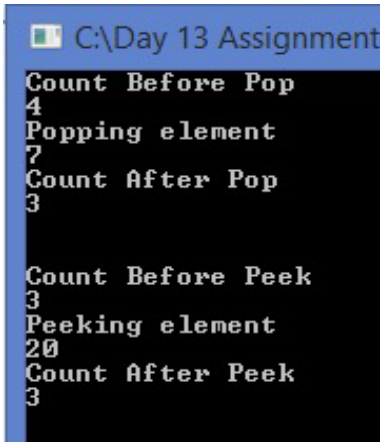
11. WACP to illustrate usage of Stack<>. Write couple of points about Stack

Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```

```
using System.Threading.Tasks;

namespace Day_13_Project_9
{
    // Author : Praveen Chakravarthi
    // Purpose : Stack
    internal class Program
    {
        static void Main(string[] args)
        {
            Stack<int> data = new Stack<int>();
            data.Push(12);
            data.Push(23);
            data.Push(20);
            data.Push(7);

            // Popping the Values
            Console.WriteLine("Count Before Pop");
            Console.WriteLine(data.Count);
            Console.WriteLine("Popping element");
            Console.WriteLine(data.Pop());
            Console.WriteLine("Count After Pop");
            Console.WriteLine(data.Count);

            // Peeking the Values
            Console.WriteLine("\n");
            Console.WriteLine("Count Before Peek");
            Console.WriteLine(data.Count);
            Console.WriteLine("Peeking element");
            Console.WriteLine(data.Peek());
            Console.WriteLine("Count After Peek");
            Console.WriteLine(data.Count);
            Console.ReadLine();


        }
    }
}
```

Output:



About Stack:

Stack represents First in- Last out/ Last in-First out collection of object

- To add items in to the list ‘Push’ is the keyword used
- To remove the items from the list ‘Pop’ is the keyword used
- ‘Peek’ keyword is used to return the top most item in the list but doesn’t remove it

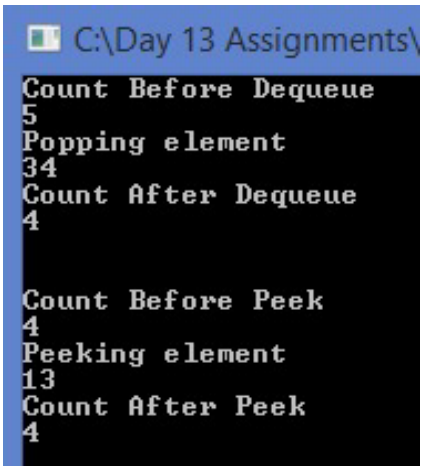
12. WACP to illustrate usage of Queue<>. Write couple of points about Stack

Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Day_13_Project_10
{
    internal class Program
    {
        // Author : Praveen Chakravarthi
        // Purpose : Queue
        static void Main(string[] args)
        {
            Queue<int> data = new Queue<int>();
            data.Enqueue(34);
            data.Enqueue(13);
            data.Enqueue(9);
            data.Enqueue(79);
            data.Enqueue(62);

            // Dequeueing the Values
            Console.WriteLine("Count Before Dequeue");
            Console.WriteLine(data.Count);
            Console.WriteLine("Popping element");
            Console.WriteLine(data.Dequeue());
            Console.WriteLine("Count After Dequeue");
            Console.WriteLine(data.Count);

            // Peeking the Values
            Console.WriteLine("\n");
            Console.WriteLine("Count Before Peek");
            Console.WriteLine(data.Count);
            Console.WriteLine("Peeking element");
            Console.WriteLine(data.Peek());
            Console.WriteLine("Count After Peek");
```

```
            Console.WriteLine(data.Count);
            Console.ReadLine();
        }
    }
}
```

Output:



About Queue:

Queue represents First in- First out collection of object

- To add items in to the list we use ‘Enqueue’ keyword
- To remove items from the list we use ‘Dequeue’
- ‘Peek’ keyword is used to return the top most item in the list but doesn’t remove it