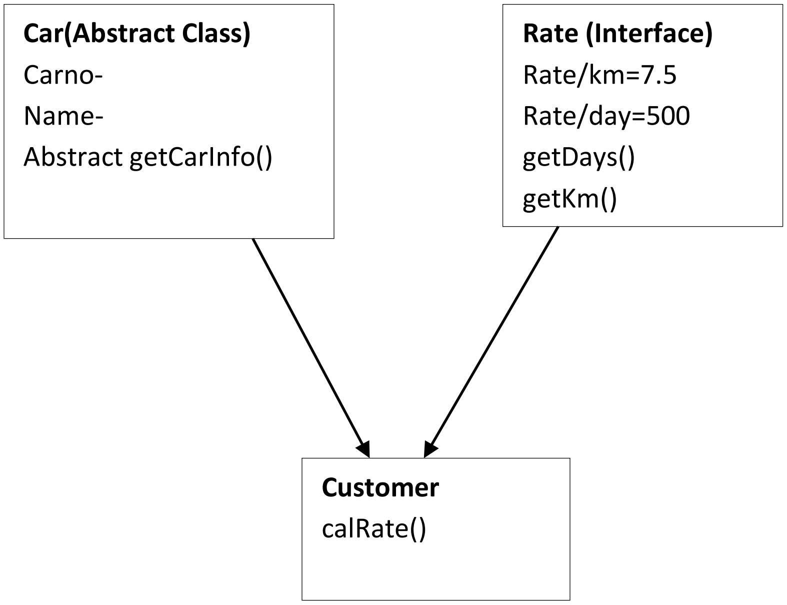
Interface and multithreading

1. Write a program to perform following interface.



Car(Abstract Class)
Carno-
Name-
Abstract getCarInfo()

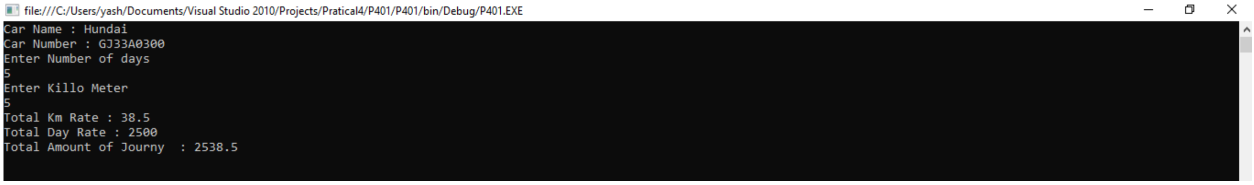
Rate (Interface)
Rate/km=7.5
Rate/day=500
getDays()
getKm()

Customer
calRate()

Display all info about Customer, Car, Km, Days and Total Amount.

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

namespace P401
{
    public abstract class CarDetails
    {
        public String carname;
        public String carnumber;
       abstract public void Getcarinfo(string carname, string carnumber);
    }
    public interface RateDetails
    {
        void getdays();
        void getkm();
    }

    class Program : CarDetails, RateDetails
    {
        private float rate_km = 7.7f;
        private float rate_day = 500;
        public int getday;
        public int getkm;

        public override void Getcarinfo(string carname,
        string carnumber)
        {
            String Carname = carname;
            String Carnumber = carnumber;
             Console.WriteLine("Car Name : {0} ",
             Carname);
            Console.WriteLine("Car Number : {0} ",
            Carnumber);
        }

        public void get_day()
        {
            Console.WriteLine("Enter Number of days");
             getday =
             Convert.ToInt32(Console.ReadLine());
            //Console.WriteLine(rate_day);
        }

        public void get_km()
        {
            Console.WriteLine("Enter Killo Meter");
            getkm = Convert.ToInt32(Console.ReadLine());
        }
        public void getdays()
        {
            throw new NotImplementedException();
        }

        void RateDetails.getkm()
        {
            throw new NotImplementedException();
        }

        public void calrate()
        {
            float ratekm = rate_km * getkm;
            float rateday = rate_day * getday;
            float total = ratekm + rateday;
             Console.WriteLine("Total Km Rate : {0} ",
             ratekm);
            Console.WriteLine("Total Day Rate : {0} ",
            rateday);
            Console.WriteLine("Total Amount of Journy  :
            {0} ", total);
        }
        static void Main(string[] args)
        {

            Program c = new Program();
            c.Getcarinfo("Hundai", "GJ33A0300");
            c.get_day();
            c.get_km();
            c.calrate();
            Console.ReadKey();
        }
    }
}
```

O/P:-


file:///C:/Users/yash/Documents/Visual Studio 2010/Projects/Pratical4/P401/P401/bin/Debug/P401.EXE

```
Car Name : Hundai
Car Number : GJ33A0300
Enter Number of days
5
Enter Killo Meter
5
Total Km Rate : 38.5
Total Day Rate : 2500
Total Amount of Journy  : 2538.5
```

2.W.A.P. having two threads one thread display Alphabets @ every 2 seconds and another thread display numbers from 1 to 20 @ every 1 second.

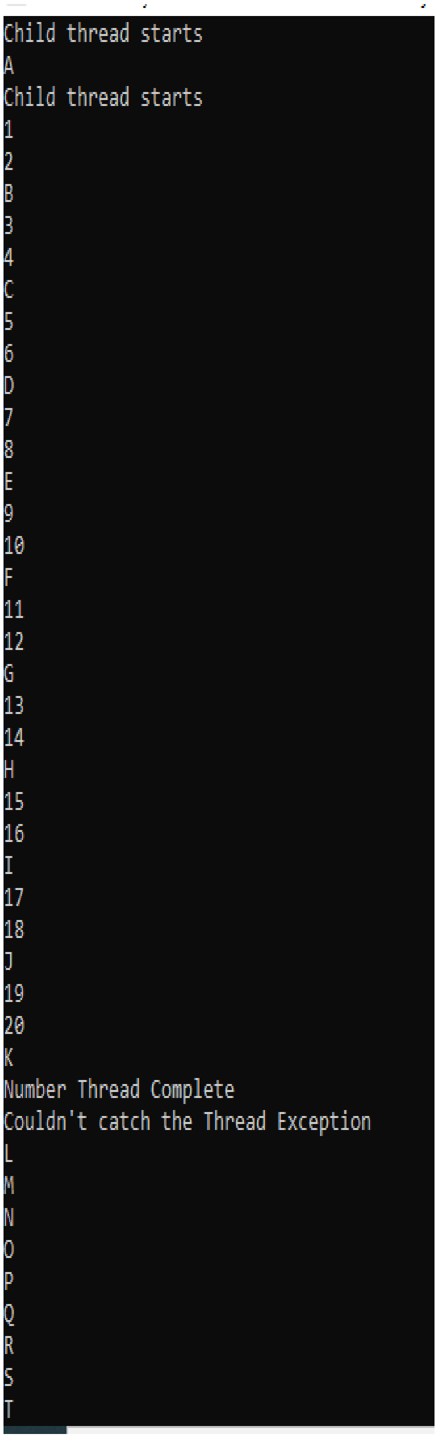
```
using System;
using System.Collections.Generic;
using System.Text;
using System.Threading;

namespace ThreadDemo
{
class Program
{
    public static void CallToChildThread()
{
try
{

Console.WriteLine("Child thread starts");
// do some work, like counting to 10
for (char i ='A'; i <= 'Z'; i++)
{
Console.WriteLine(i);
Thread.Sleep(2000);
}
Console.WriteLine("Alphabeat Thread Complete");
```

```
}
catch (ThreadAbortException e)
{
Console.WriteLine("Thread Abort Exception");
}
finally
{
Console.WriteLine("Couldn't catch the Thread
Exception");
}

}
    public static void Call()
    {
        try
        {

            Console.WriteLine("Child thread starts");
            // do some work, like counting to 10
            for (int i = 1; i <= 20; i++)
            {
                Console.WriteLine(i);
                Thread.Sleep(1000);
            }
            Console.WriteLine("Number Thread Complete");

        }
        catch (ThreadAbortException e)
        {
            Console.WriteLine("Thread Abort Exception");
        }
        finally
        {
            Console.WriteLine("Couldn't catch the Thread
Exception");
```

```
        }

    }

static void Main(string[] args)
{
ThreadStart childref = new
ThreadStart(CallToChildThread);
Thread childThread = new Thread(childref);
ThreadStart ch= new ThreadStart(Call);
Thread child= new Thread(ch);
childThread.Start();
child.Start();
Console.ReadKey();
}

}
}
```

Interface and multithreading

O/P:-



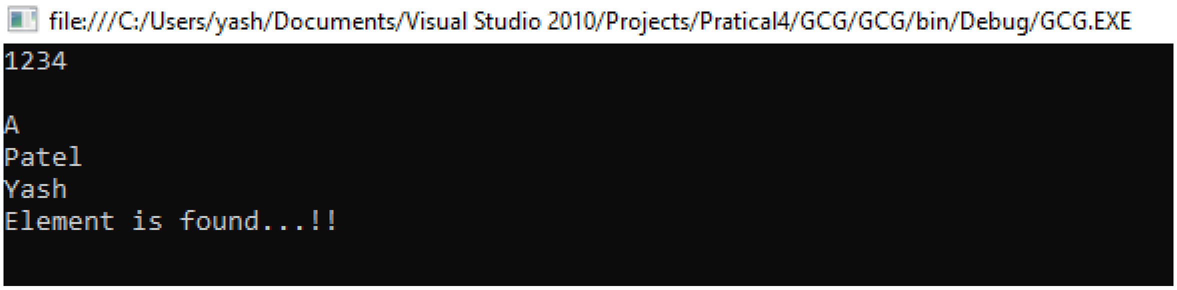
```
Child thread starts
A
Child thread starts
1
2
B
3
4
C
5
6
D
7
8
E
9
10
F
11
12
G
13
14
H
15
16
I
17
18
J
19
20
K
Number Thread Complete
Couldn't catch the Thread Exception
L
M
N
O
P
Q
R
S
T
```

3 Write a c# program to perform stack operation.

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

namespace GCG
{
    class GCG
    {
        static void Main(string[] args)
        {
            Stack my_stack = new Stack();
            my_stack.Push("Yash");
            my_stack.Push("Patel");
            my_stack.Push('A');
            my_stack.Push(null);
            my_stack.Push(1234);
            my_stack.Push(490.98);
            my_stack.Pop();
            foreach (var elem in my_stack)
            {
                Console.WriteLine(elem);
            }
              if (my_stack.Contains(1234) == true)
             {
               Console.WriteLine("Element is
               found...!!");
             }

        else
        {
```

```
                    Console.WriteLine("Element is not
                    found...!!");
               }
                     Console.ReadKey();
               }
          }
     }
```

O/P:-

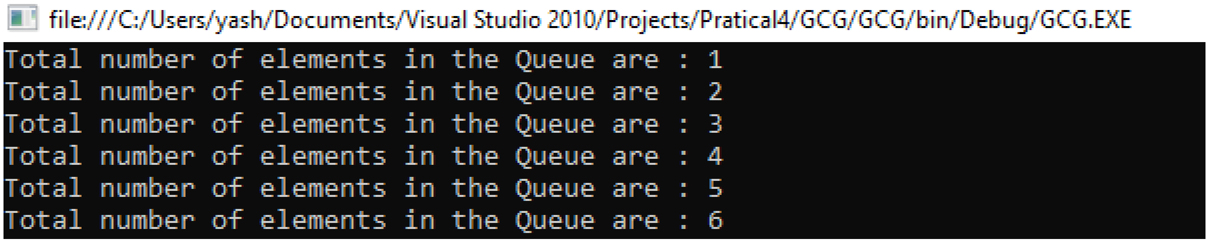


file:///C:/Users/yash/Documents/Visual Studio 2010/Projects/Pratical4/GCG/GCG/bin/Debug/GCG.EXE

```
1234

A
Patel
Yash
Element is found...!!
```

4 Write a c# program to perform queue operation.

```
using System;
using System.Collections;

class GFG
{
    public static void Main()
    {

        Queue myQueue = new Queue();
        myQueue.Enqueue("one");
         Console.Write("Total number of elements in the
         Queue are : ");
        Console.WriteLine(myQueue.Count);
        myQueue.Enqueue("two");
         Console.Write("Total number of elements in the
         Queue are : ");
        Console.WriteLine(myQueue.Count);
        myQueue.Enqueue("three");
         Console.Write("Total number of elements in the
         Queue are : ");
        Console.WriteLine(myQueue.Count);
        myQueue.Enqueue("four");
         Console.Write("Total number of elements in the
         Queue are : ");
        Console.WriteLine(myQueue.Count);
        myQueue.Enqueue("five");
         Console.Write("Total number of elements in the
         Queue are : ");
        Console.WriteLine(myQueue.Count);
        myQueue.Enqueue("six");
         Console.Write("Total number of elements in
         the Queue are : ");

        Console.WriteLine(myQueue.Count);
        Console.ReadKey();
    }
}
O/P:-
```



file:///C:/Users/yash/Documents/Visual Studio 2010/Projects/Pratical4/GCG/GCG/bin/Debug/GCG.EXE

```
Total number of elements in the Queue are : 1
Total number of elements in the Queue are : 2
Total number of elements in the Queue are : 3
Total number of elements in the Queue are : 4
Total number of elements in the Queue are : 5
Total number of elements in the Queue are : 6
```

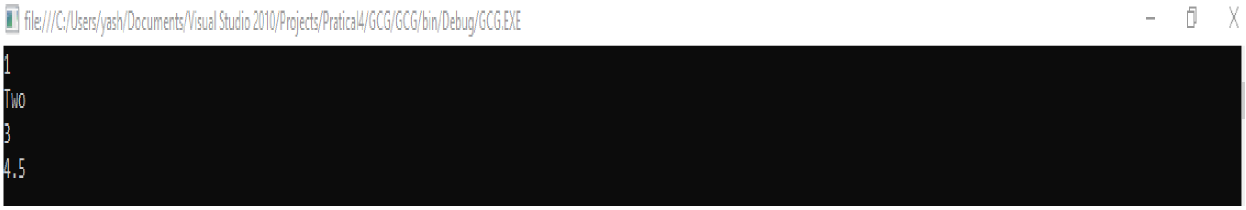
5 Write a c# program to perform array List.

```
using System;
using System.Collections;

class GFG
{
    public static void Main()
    {
```

Interface and multithreading

```
        ArrayList myArryList = new ArrayList();
        myArryList.Add(1);
        myArryList.Add("Two");
        myArryList.Add(3);
        myArryList.Add(4.5f);
        int firstElement = (int)myArryList[0];
        string secondElement = (string)myArryList[1];
        int thirdElement = (int)myArryList[2];
        float fourthElement = (float)myArryList[3];

        myArryList.Remove(100);

        foreach (var item in myArryList)
            Console.WriteLine(item);
        Console.ReadKey();


    }
}
```

O/P:-


file:///C:/Users/yash/Documents/Visual Studio 2010/Projects/Pratical4/GCG/GCG/bin/Debug/GCG.EXE

```
1
Two
3
4.5
```