Assignment Day 4 – Rahul Kumar

Interface Program 1

```
namespace Interface1.Shift
{
    public interface IShifts
    {
         int work();
    }
}
```

```
using Interface1.Shift;

namespace Interface1.Employee
{
    public class Employees : IShifts
    {
        public int work()
        {
            return 9;
        }
    }
}
```

```
using System;
using Interface1.Shift;
using Interface1.Employee;
namespace Interface1
{
    class Program
    {
        static void Main(string[] args)
        {
            var obj = new Employees();
            int temp = obj.work();
            Console.WriteLine(temp);
        }
    }
}
```

```
C:\Users\Rahul_7k7\Desktop\TaazaaAssignments\Assign4\Interface1>dotnet run
9
```

Interface Program 2 – Multiple Interface

```
namespace InterfaceMultiple.Rules
{
    public interface IAgreement
    {
         string agreement();
    }
}
```

```
namespace InterfaceMultiple.Rules
{
    public interface ILeaves
    {
         int leaves();
    }
}
```

```
namespace InterfaceMultiple.Rules
{
    public interface IShifts
    {
         int work();
    }
}
```

```
namespace InterfaceMultiple.Rules
{
    public interface ISalary
    {
         int salary();
    }
}
```

```
using InterfaceMultiple.Rules;

namespace InterfaceMultiple.Employee
{
    public class Employees : IAgreement, ILeaves, ISalary, IShifts
    {
        public string agreement()
        {
            return "You have to serve 1 month notice period";
        }

        public int leaves()
        {
            return 15;
        }

        public int salary()
        {
            return 400000;
        }

        public int work()
        {
            return 9;
        }
    }
}
```

```
using System;
using InterfaceMultiple.Rules;
using InterfaceMultiple.Employee;


namespace InterfaceMultiple
{
    class Program
    {
        static void Main(string[] args)
        {
            var obj = new Employees();
            string agree = obj.agreement();
            int lev = obj.leaves();
            int wrk = obj.work();
            int sal = obj.salary();
            Console.WriteLine(agree + "\nNo of leaves: " + lev +"\nWorking Hou
rs: "+ wrk+ "\nYour Salary: " + sal);
        }
    }
}
```

```
C:\Users\Rahul_7k7\Desktop\TaazaaAssignments\Assign4\InterfaceMultiple>dotnet run

You have to serve 1 month notice period

No of leaves: 15

Working Hours: 9

Your Salary: 400000
```

Interface Problem 3

```
using System;

namespace Interface3
{
    //interface declaration
    interface Broker
    {
        int Rent();
        string Buy();
        string Sell();
    }
    class Buyer : Broker
    {
        public string Buy()
        {
            return "40 Lakhs";
        }

        public int Rent()
        {
            return 8500;
        }

        public string Sell()
        {
            return null;
        }
    }
    class Seller : Broker
    {
        public string Buy()
        {
            return null;
        }

        public int Rent()
        {
            return 7500;
        }

        public string Sell()
        {
            return "25 Lakhs";
        }
    }

    class Program
    {
        static void Main(string[] args)
        {
            var obj1 = new Buyer();
            string b1 = obj1.Buy();
            int r1 = obj1.Rent();
            Console.WriteLine("Buyer:\n"+"Can buy at: "+ b1 + "\nCan Rent at:
"+ r1);
            var obj2 = new Seller();
            string s2 = obj2.Sell();
            int r2 = obj2.Rent();
            Console.WriteLine("Seller:\n"+"Can sell at: "+ s2 + "\nCan Rent at
: "+ r2);
        }
    }
}
```

```
C:\Users\Rahul_7k7\Desktop\TaazaaAssignments\Assign4\Interface3>dotnet run

Buyer:

Can buy at: 40 Lakhs

Can Rent at: 8500

Seller:

Can sell at: 25 Lakhs

Can Rent at: 7500
```

Abstract Class Program 1 – To calculate Perimeter of Rectangle

```
using System;

namespace AbstractCl1
{
    abstract class PerimeterClass
    {
        public abstract int Perimeter();
    }

    class Rectangle : PerimeterClass
    {
        int len = 0;
        int br = 0;
        public Rectangle(int l, int b)
        {
            len = l;
            br = b;
        }

        public override int Perimeter()
        {
            int sum = len+br;
            return 2*sum;
        }
    }
    class Program
    {
        static void Main(string[] args)
        {
            var obj = new Rectangle(5, 2);
            Console.WriteLine("Perimeter: "+ obj.Perimeter());
        }
    }
}
```

```
C:\Users\Rahul_7k7\Desktop\TaazaaAssignments\Assign4\AbstractCl1>dotnet run

Perimeter: 14
```

Abstract Class Problem 2

```
using System;

namespace AbstractCl2
{
    abstract class AbClass      //abstract class
    {
        //Non Abstract Method in abstrct class
        public int AddNumbers(int num1, int num2)
        {
            return num1+num2;
        }
        //Abstract Method in abstract class
        public abstract int SubNumbers(int num1, int num2);
    }

    //child class of abstract class
    class Derived : AbClass
    {
        // implementing the abstract method
        public override int SubNumbers(int num1, int num2)
        {
            return num1-num2;
        }
    }
    class Program
    {
        static void Main(string[] args)
        {
            var obj = new Derived();
            Console.WriteLine("Addition: "+ obj.AddNumbers(8, 5)+"\nSubstracti
on: "+ obj.SubNumbers(12, 8));
        }
    }
}
```

```
C:\Users\Rahul_7k7\Desktop\TaazaaAssignments\Assign4\AbstractCl2>dotnet run

Addition: 13

Substraction: 4
```

Constructor Chaining

```
using System;

namespace ConstructorChaining
{
    public class ConstructMe
    {
        public string name;
        public int id;
        public ConstructMe(string Name, int id)
        {
            this.name = Name;
            this.id = id;
        }

    }
    class program
    {
        static void Main(string[] args)
        {
            ConstructMe a = new ConstructMe("Rahul", 200);
            Console.WriteLine("Name: "+ a.name+"\n"+"ID: "+a.id);
        }
    }
}
```

```
C:\Users\Rahul_7k7\Desktop\TaazaaAssignments\Assign4\ConstructorChaining>dotnet run

Name: Rahul

ID: 200
```

Enum

```
using System;

namespace Enum
{
    class Program
    {
        enum Months
        {
            Null,       // 0
            January,    // 1
            February,   // 2
            March,      // 3
            April,      // 4
            May,        // 5
        }
        static void Main(string[] args)
        {
            int obj1 = (int) Months.May;
            Console.WriteLine(obj1);
        }
    }
}
```

```
C:\Users\Rahul_7k7\Desktop\TaazaaAssignments\Assign4\Enum>dotnet run
```

```
5
```

Struct

```csharp
using System;

struct Employees {
   public string Name;
   public string Email;
   public int Id;
};

public class testStructure {
   public static void Main(string[] args) {
      Employees Emp1;    // Declare Emp1
      Employees Emp2;

      // emp 1 specification
      Emp1.Name = "Rahul";
      Emp1.Email = "abc@abc.com";
      Emp1.Id = 572;

      // emp 2 specification
      Emp2.Name = "Harpreet";
      Emp2.Email = "bbc@abc.com";
      Emp2.Id = 957;

      Console.WriteLine( "Name : {0}", Emp1.Name);
      Console.WriteLine("Email : {0}", Emp1.Email);
      Console.WriteLine("ID :{0}", Emp1.Id);

      Console.WriteLine( "Name : {0}", Emp2.Name);
      Console.WriteLine("Email : {0}", Emp2.Email);
      Console.WriteLine("ID :{0}", Emp2.Id);
   }
}
```

```
C:\Users\Rahul_7k7\Desktop\TaazaaAssignments\Assign4\Struct>dotnet run

Name : Rahul

Email : abc@abc.com

ID :572

Name : Harpreet

Email : bbc@abc.com

ID :957
```