DAY 11 ASSIGNMENT

(BY G V S S SRI LASYA)

1) Write the 7 points discussed about properties.

- Properties are same as class variables with **get;** and **set;**
- Properties are created to access private variables of the class
- They help to regulate assignment of values to variables.Else,when made public,variables can be assigned values any number of times.
- Unlike normal variables,names of properties should start with **an upper case letter.**
- We can make computations within a property.
- Property with only **get{;}** is **read only**
- Property with only **set{;}** is **write only**
- Property with both **get{;}** and **set{;}** can be **read and written**
- A simple example of properties is

```
class Employee
    {
        private int id;
        private string name;
        private string designation;
        private string salary;

        public int Id
        {
            get { return id; }
            set { id = value; }
        }
    }
```

2) Write the 6 points about interface discussed in the class.

- Interface is like pure abstract class
- Name of an interface should start with “I”
- Interface acts like a contract which has to be fulfilled.
- Any class which implements the interface should necessarily override all of its methods
- By default, all of the methods in interface are public and abstract
- Interface supports multiple inheritance

3) Research and write the difference between abstract class and interface in C#

Abstract class	**Interface**
1) Can have non abstract memebers too	Can only have abstract methods
2)Declared using “abstract class”	Declared using “interface InameOfInterface”
3)Can have fields	Can not have fields
4)Allows access modifiers to make the members public or private	By default all the member methods are public and cant be chaneged
5)Doesn’t allow multiple inheritance	Allows multiple inheritance
6)Can inherit and be inherited from another abstract class or interface	Can be inherited and be inherited from other interfaces only.

4. Research and understand when to create static methods.

- Method is not using any instance variables(can be dealing with no variables or static variables only)
- When that method has to be commonly shared by all instances of the class
- When definition of the method should not be changed or overridden
- While writing utility classes which can have only static methods as their members

5) Write example program for interface Ishape discussed in the class Ishape.Include the classes Cricle, Square, Triangle, Rectangle

CODE

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

/*********************************************************************
Author : G V S S SRI LASYA
Purpose: Write example program for interfaces discussed in the class.
Create IShape interface and include the classes
Cricle, Square, Triangle, Rectangle.
*********************************************************************/



namespace day11project1
{
    interface IShape
    {
        int CalculatePerimeter();
        int CalculateArea();

    }

    class Circle : IShape
    {

        private int radius;

        /// <summary>
        /// Read user input for radius
        /// </summary>
        /// <return>
        /// void
        /// </return>
        public void ReadRadius()
        {

            Console.Write("\n\n\n\nEnter the radius : ");
            radius = Convert.ToInt32(Console.ReadLine());

        }


        /// <summary>
        /// Calculate area of circle
        /// </summary>
        /// <returns>
        /// int
        /// </returns>
        public int CalculateArea()
        {

            return (int)22 * radius * radius / 7;

        }


        /// <summary>
        /// Calculate perimeter of circle
        /// </summary>
        /// <returns>
        /// int
        /// </returns>
        public int CalculatePerimeter()
        {

            return (int)2 * 22 * radius / 7;
```

```
        }

    }

    class Square : IShape
    {
        private int side;

        /// <summary>
        /// Read user input for side of square
        /// </summary>
        ///<return>
        ///void
        /// </return>
        public void ReadSide()
        {

            Console.Write("\n\n\n\nEnter the side of square : ");
            side = Convert.ToInt32(Console.ReadLine());

        }


        /// <summary>
        /// Calculate perimeter of square
        /// </summary>
        ///<return>
        ///int
        /// </return>
        public int CalculatePerimeter()
        {

            return 4 * side;

        }


        /// <summary>
        /// Calculate area of square
        /// </summary>
        ///<return>
        ///int
        /// </return>
        public int CalculateArea()
        {

            return side * side;

        }
    }

    class Rectangle : IShape
    {
        private int length, breadth;


        /// <summary>
        /// Read user input for sides of rectangle
        /// </summary>
        ///<return>
        ///void
        /// </return>
        public void ReadSides()
        {
            Console.Write("\n\n\n\nEnter length of the rectangle : ");
            length = Convert.ToInt32(Console.ReadLine());
            Console.Write("Enter breadth of the rectangle : ");
            breadth = Convert.ToInt32(Console.ReadLine());

        }

        /// <summary>
        /// Calculate perimeter of rectangle
        /// </summary>
```

```
        ///<return>
        ///int
        /// </return>
        public int CalculatePerimeter()
        {
            return 2 * (length + breadth);
        }

        /// <summary>
        /// Calculate area of square
        /// </summary>
        ///<return>
        ///int
        /// </return>
        public int CalculateArea()
        {
            return length * breadth;
        }

    }

    class EquilateralTriangle : IShape
    {

        private int side;



        /// <summary>
        /// Read user inut for side of equilaterak triangle
        /// </summary>
        ///<return>
        ///void
        /// </return>
        public void ReadSide()
        {
            Console.Write("\n\n\n\nEnter side of the equilateral triangle : ");
            side = Convert.ToInt32(Console.ReadLine());
        }

        /// <summary>
        /// Calculate perimeter of equilateral triangle
        /// </summary>
        ///<return>
        ///int
        /// </return>
        public int CalculatePerimeter()
        {
            return 3 * side;
        }

        /// <summary>
        /// Calculate area of square
        /// </summary>
        ///<return>
        ///int
        /// </return>
        public int CalculateArea()
        {
            return (int)1.732 * side * side / 4;

        }

    }
    internal class Program
    {
        static void Main(string[] args)
        {
            Console.Write("\n\nCalculating areas and perimeters of
circle,square,rectangle,equilateral triangle");

            Circle circleObject = new Circle();
            circleObject.ReadRadius();
            Console.Write("Area of circle : " + circleObject.CalculateArea());
```

```
            Console.Write("\nPerimeter of circle : " + circleObject.CalculatePerimeter());


            Square squareObject = new Square();
            squareObject.ReadSide();
            Console.Write($"Area of square : " + squareObject.CalculateArea());
            Console.Write($"\nPerimeter of square : " + squareObject.CalculatePerimeter());


            Rectangle rectangleObject = new Rectangle();
            rectangleObject.ReadSides();
            Console.Write($"Area of rectangle : " + rectangleObject.CalculateArea());
            Console.Write($"\nPerimeter of rectangle : " +
rectangleObject.CalculatePerimeter());


            EquilateralTriangle triangleObject = new EquilateralTriangle();
            triangleObject.ReadSide();
            Console.Write($"Area of triangle : " + triangleObject.CalculateArea());
            Console.Write($"\nPerimeter of triangle : " +
triangleObject.CalculatePerimeter());


            Console.ReadLine();

        }
    }
}
```

OUTPUT

```
Calculating areas and perimeters of circle,square,rectangle,equilateral triangle



Enter the radius : 3
Area of circle : 28
Perimeter of circle : 18



Enter the side of square : 4
Area of square : 16
Perimeter of square : 16



Enter length of the rectangle : 5
Enter breadth of the rectangle : 4
Area of rectangle : 20
Perimeter of rectangle : 18



Enter side of the equilateral triangle : 5
Area of triangle : 6
Perimeter of triangle : 15
```

6) Create a class Employee with only properties.

CODE

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

/*************************************************************************
Author : G V S S SRI LASYA
Purpose :  Create a class Employee with only properties.
*************************************************************************/


namespace Day11Project2
{

    class Employee
    {
        public int Id { get; set; }
        public string Name { get; set; }
        public int Salary
        {
            get
            {
                return (Designation == "HR") ? (200000) : (100000);

            }

        }

        public string Designation { get; set; }
    }

    internal class Program
    {
        static void Main(string[] args)
        {
            Employee emp1 = new Employee();

            //setting properties
            emp1.Id = 1;
            emp1.Name = "Juhi";
            emp1.Designation = "Software Trainee";


            Console.Write("\nPrinting employee details");

            //getting properties
            Console.Write("\n\n\n\nEmployee Id : " +  emp1.Id);
            Console.Write("\n\nEmployee name : " + emp1.Name);
            Console.Write("\n\nEmployee designation : " + emp1.Designation);
            Console.Write("\n\nEmployee salary : " + emp1.Salary);

            Console.ReadLine();

        }
    }
}
```

OUTPUT

```
Printing employee details


Employee Id : 1

Employee name : Juhi

Employee designation : Software Trainee

Employee salary : 100000
```

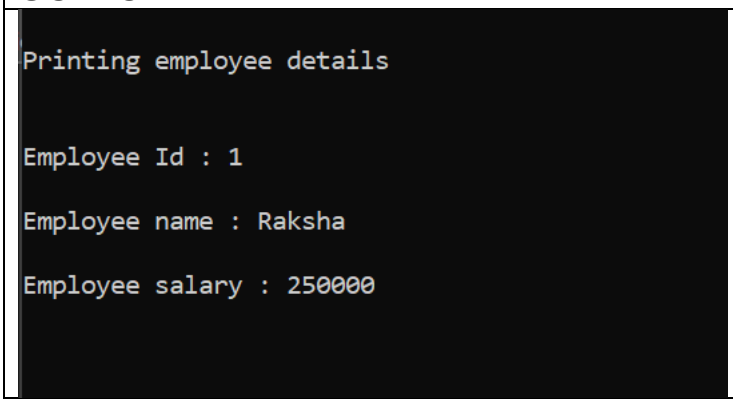
7) Write sample code to illustrate properties as discussed in class.
id-get, set
name-get,set
designation-set (writeonly)
salary-get (get with some functionality)

CODE

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

/*********************************************************************
Author : G V S S SRI LASYA
Purpose: Write sample code to illustrate properties as discussed in class.
id-get, set
name-get,set
designation-set (writeonly)
salary-get (get with some functionality)
*********************************************************************/


namespace day11project3
{
    public class Employee
    {
        private int id;
        private string name;
        private string designation;
        private int salary;


        public int Id
        {
            set { id = value; }
            get { return id; }
        }

        public string Name
        {
            set { name = value; }
            get { return name; }
        }

        public string Designation
        {
            set { designation = value; }
        }

        public int Salary
        {
            get
            {
                salary = (designation == "HR") ? 250000 : 150000;
                return salary;
            }
        }

    }




    internal class Program
    {
        static void Main(string[] args)
        {
            Employee emp1 = new Employee();

            //setting properties
```

```
            emp1.Id = 1;
            emp1.Name = "Raksha";
            emp1.Designation = "HR";

            //getting properties
            Console.Write("\n\nPrinting employee details");
            Console.Write("\n\n\nEmployee Id : " + emp1.Id);
            Console.Write("\n\nEmployee name : " + emp1.Name);
            Console.Write("\n\nEmployee salary : " + emp1.Salary);

            Console.ReadLine();




        }
    }
}
```

OUTPUT



8) Create Mathematics class and add 3 static methods and call the methods in main method.

CODE

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

/*************************************************************************
Author : G V S S SRI LASYA
Purpose : Create Mathematics class and add 3 static methods and call the
methods in main method.
*************************************************************************/

namespace day11project4
{
    class Maths
    {
        /// <summary>
        /// adding 2 integers and returning sum
        /// </summary>
        /// <param name="a"></param>
        /// <param name="b"></param>
        /// <returns>
        /// int
        /// </returns>
        public static int Add(int a,int b)
        {
            return a + b;
        }


        /// <summary>
        /// subtracting 2 integers and returning difference
        /// </summary>
        /// <param name="a"></param>
        /// <param name="b"></param>
        /// <returns>
        /// int
        /// </returns>
        public static int Subtract(int a, int b)
        {
            return a - b;
        }

        /// <summary>
        /// multiplying 2 integers and returning product
        /// </summary>
        /// <param name="a"></param>
        /// <param name="b"></param>
        /// <returns>
        /// int
        /// </returns>
        public static int Multiply(int a,int b)
        {
            return a * b;
        }
    }
    internal class Program
    {
        static void Main(string[] args)
        {
            int a, b;

            Console.Write("\n\nPrinting sum,difference,product of 2 integers");

            //taking user inputs
            Console.Write("\n\n\n\nEnter first integer : ");
            a = Convert.ToInt32(Console.ReadLine());
            Console.Write("\nEnter second integer : ");
            b = Convert.ToInt32(Console.ReadLine());
```

```
            //printing outputs
            Console.Write("\n\n\n\nSum of " + a + " and " + b + " : " + Maths.Add(a, b));
            Console.Write("\n\nDifference of " + a + " and " + b + " : " + Maths.Subtract(a,
b));
            Console.Write("\n\nProduct of " + a + " and " + b + " : " + Maths.Multiply(a,
b));

            Console.ReadLine();
        }
    }
}
```

OUTPUT

```
Printing sum,difference,product of 2 integers



Enter first integer : 45

Enter second integer : 3




Sum of 45 and 3 : 48

Difference of 45 and 3 : 42

Product of 45 and 3 : 135
```