**Lesson03 Liskov Substitution Principle**

**Notes:-**

**1-Liskov substitution is called about substitute base type from sub type**

**By using virtual and override principle which allow instance type to access to its properties on the base class as below**

**Example:-**

**using System;**

**namespace LiskovoSubsitutionPro{**

**//we see that the width and height access on the base class only**

**public class Rectangle{**

**public int Width { get; set; }**

**public int Height { get; set; }**

**public Rectangle(){}**

**public Rectangle(int width, int height){Width = width;Height = height;}**

**public override string ToString(){return $"{nameof(Width)}: {Width}, {nameof(Height)}: {Height}";}}**

**//we see that the sub class properties using new prosperity**

**public class Square : Rectangle{**

**public new int Width{set { base.Width = base.Height = value; }}**

**public new int Height{set { base.Width = base.Height = value; }}}**

**public class Demo{**

**//we can define static method as variable as below**

**static public int Area(Rectangle r) => r.Width \* r.Height;**

**static void Main(string[] args){**

**Rectangle rc = new Rectangle(2, 3);**

**Console. WriteLine($"{rc} has area {Area(rc)}");**

**//we see that width and height it apply only on the parent class level**

**Rectangle sq = new Square();**

**sq.Width = 4;**

**Console.WriteLine($"{sq} has area {Area(sq)}");}}}**

**With using Liskovo Principle**

**using System;**

**namespace LiskovoSubsitutionPro{**

**// using a classic example**

**public class Rectangle{**

**public virtual int Width { get; set; }**

**public virtual int Height { get; set; }**

**public Rectangle(){}**

**public Rectangle(int width, int height){**

**Width = width;Height = height;}**

**public override string ToString(){return $"{nameof(Width)}: {Width}, {nameof(Height)}: {Height}";}}**

**public class Square : Rectangle{**

**public override int Width{set { base.Width = base.Height = value; }}**

**public override int Height{set { base.Width = base.Height = value; }}}**

**public class Demo{**

**static public int Area(Rectangle r) => r.Width \* r.Height;**

**static void Main(string[] args){**

**Rectangle rc = new Rectangle(2, 3);**

**Console. WriteLine($"{rc} has area {Area(rc)}");**

**// should be able to substitute a base type for a subtype**

**Rectangle sq = new Square();**

**sq.Width = 4;**

**Console.WriteLine($"{sq} has area {Area(sq)}");}}}**