Convert the following code implementation to follow OCP

|  |
| --- |
| public class Rectangle{  public int Width { get; set; }  public int Height { get; set; }  }  public class Circle{  public double Radius { get; set; }  }  public class CombinedAreaCalculator{  public double Area(object[] shapes){  double area = 0;  foreach (var shape in shapes){  if (shape is Rectangle){  Rectangle rectangle = (Rectangle)shape;  area += rectangle.Width \* rectangle.Height;  }  if (shape is Circle){  Circle circle = (Circle)shape;  area += (circle.Radius \* circle.Radius) \* Math.PI;  }  }  return area;  }  } |

**Step1 :** Make a project and test this code is working

**Step 2:** Once you understand the code is violating open closed principle , make another project which will be refactored version of current project

**Step3:** Redesign application as given below

|  |
| --- |
| public abstract class Shape{  public abstract double Area();  }  public class Rectangle: Shape{  public int Width { get; set; }  public int Height { get; set; }  public override double Area(){  return Width \* Height;  }  }  public class Circle : Shape{  public double Radius { get; set; }  public override double Area(){  return Radius \* Radius \* Math.PI;  }  }  public class CombinedAreaCalculator{  public double Area (Shape[] shapes){  double area = 0;  foreach (var shape in shapes){  area += shape.Area();  }  return area;  }  } |