Challenge 2: Implement and Override a Method

Can you override the CalcArea() method in a derived class? A solution is placed in the solution section to help you, but we suggest you try to solve it on your own first.

WE'LL COVER THE FOLLOWING

Polymorphism in Shapes

Shapes are a perfect example of polymorphism. There are many types of shapes, e.g., circles, triangles, squares, rectangles, etc. Each of these shapes has an *area* but the way it is calculated is different for each shape. For example, a square's area will be calculated as follows:

$$SquareArea = (side)^2$$

On the other hand, the area of a circle will be calculated as follows:

$$CircleArea = pi * r^2$$

Consider we have a base class, `Shape`, and a derived class, `Circle`.

Problem Statement

Write a method in the `Circle` class which overrides the `virtual` method `CalcArea()` and returns `double` in the `Shape` class. The overriding method calculates the area of a circle and returns it.

calculates the area of a circle and returns it.

> The value of Pi is 3.14.

You are given a partially completed code in the editor. Modify the code so that the code prints the following:

Input

A radius

Output

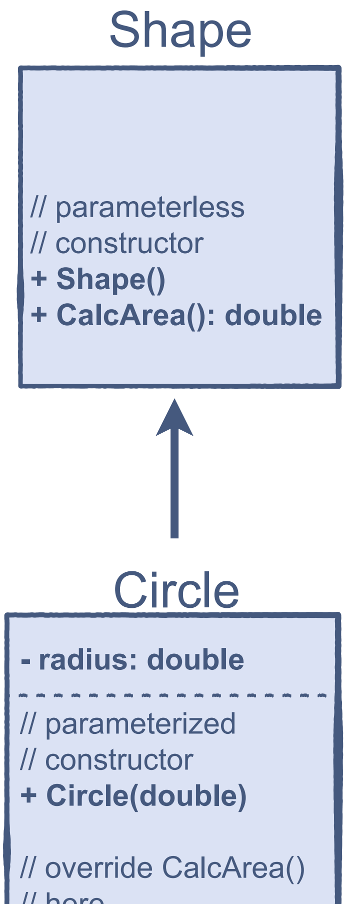
The area of the circle with the given radius

Sample Input

```
Shape circle = new Circle(2);
```

Sample Output

```
circle.CalcArea() = 12.56
```



Based and Derived Classes Structure

Coding Exercise

First, take a close look and design a step-by-step algorithm before jumping to the implementation. This problem is designed for practice, so try to solve it on your own. If you get stuck, you can always refer to the solution provided in the solution review.

Good luck!

> Uncomment **lines 4 and** 7 before writing your solution.

```
// Derived Class
class Circle : Shape {

  //private double _radius; // Uncomment this line

  public Circle(double radius) { // Parameterized Constructor
    //this._radius = radius;     // Uncomment this line
  }

  // Override the Method CalcArea() which returns the area of a Circle
  // Write your code here
}
```

The solution will be explained in the next lesson.