SD2 - Worksheet 3 - 8%

GRIFFITH COLLEGE DUBLIN

Student name:	Asralt				
Student number:	Enkhbadral				
Faculty:	Computing Science				
Course:	BSCH/BSCO/EXCH		Stage/year:	2	
Subject:	Software Development 2				
Study Mode:	Full time	✓		Part-time	
Lecturer Name:	Gemma Deery				
Assignment Title:	Worksheet 3				
Date due:					
Date submitted:					

Plagiarism disclaimer:

I understand that plagiarism is a serious offence and have read and understood the college policy on plagiarism. I also understand that I may receive a mark of zero if I have not identified and properly attributed sources which have been used, referred to, or have in any way influenced the preparation of this assignment, or if I have knowingly allowed others to plagiarise my work in this way.

I hereby certify that this assignment is my own work, based on my personal study and/or research, and that I have acknowledged all material and sources used in its preparation. I also certify that the assignment has not previously been submitted for assessment and that I have not copied in part or whole or otherwise plagiarised the work of anyone else, including other students.

Signed: _____ **Date:** _____

Please note: **Students** MUST **retain a hard / soft copy of** ALL assignments as well as a receipt issued and signed by a member of Faculty as proof of submission.
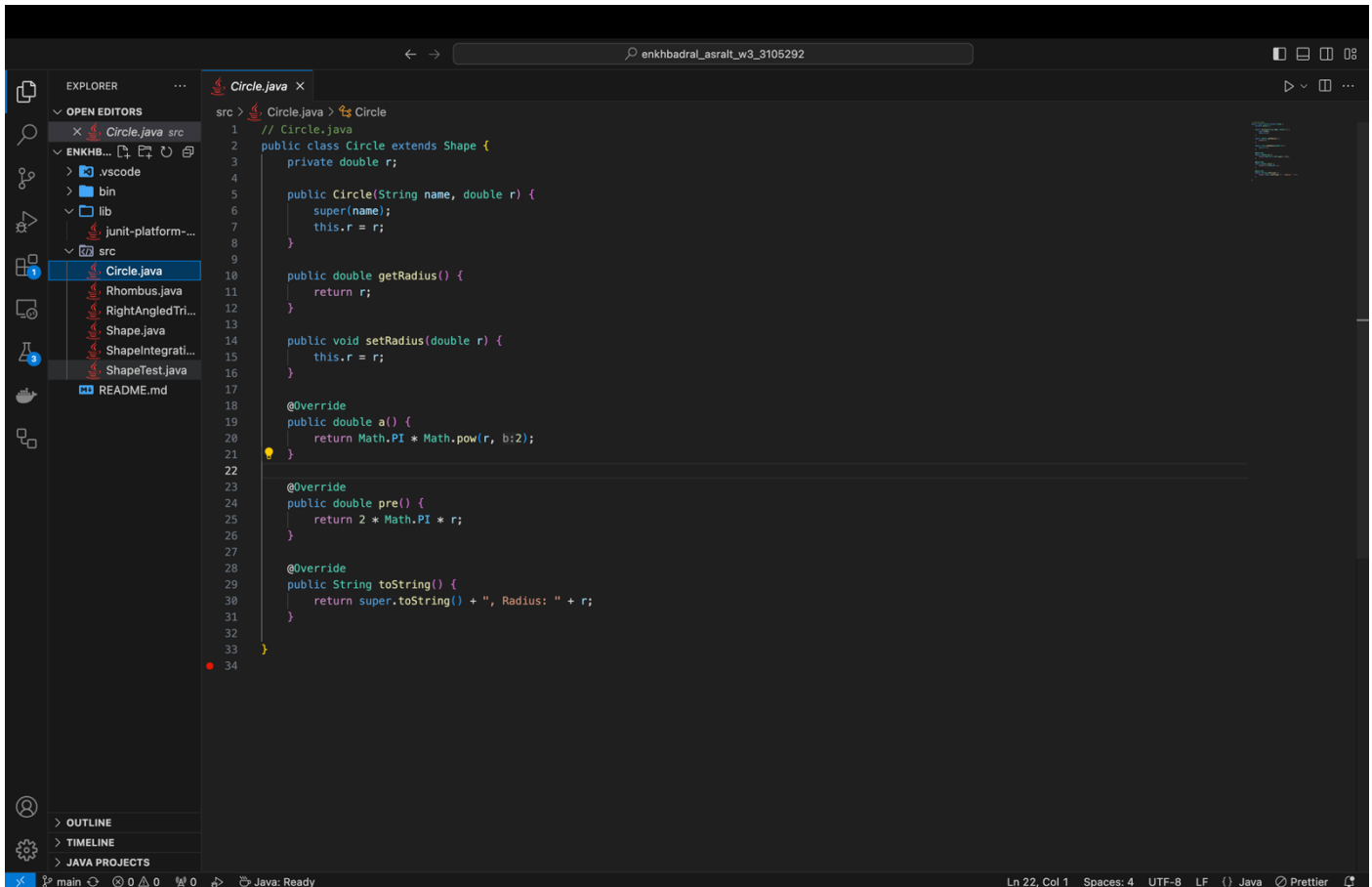
Repo Link:

Tasks

Task1.

```java
public abstract class Shape {
    private String name;

    public Shape(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public abstract double a();

    public abstract double pre();

    public String toString() {
        return "shape is " + name;
    }
}
```
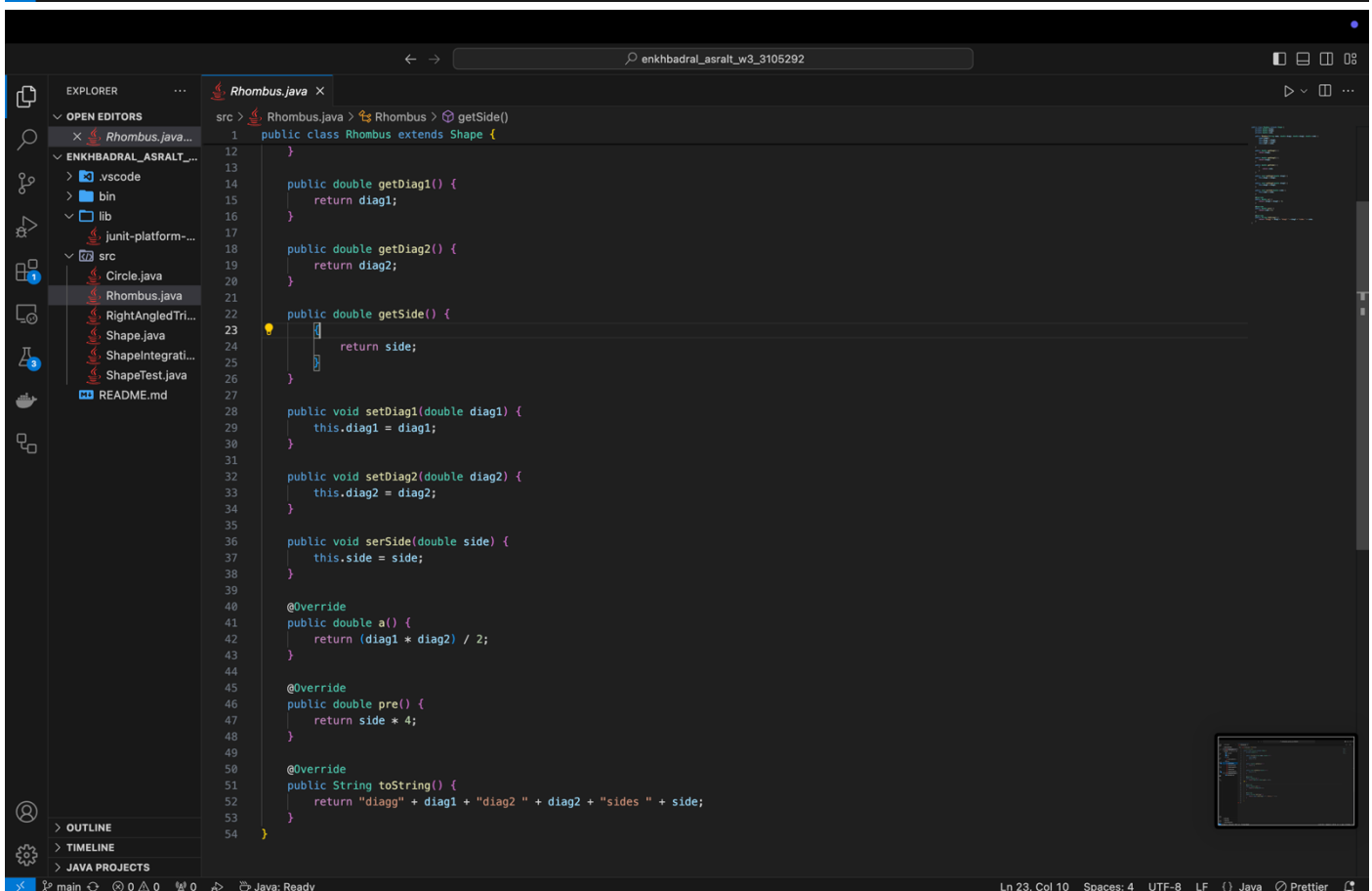
```java
// Circle.java
public class Circle extends Shape {
    private double r;

    public Circle(String name, double r) {
        super(name);
        this.r = r;
    }

    public double getRadius() {
        return r;
    }

    public void setRadius(double r) {
        this.r = r;
    }

    @Override
    public double a() {
        return Math.PI * Math.pow(r, b:2);
    }

    @Override
    public double pre() {
        return 2 * Math.PI * r;
    }

    @Override
    public String toString() {
        return super.toString() + ", Radius: " + r;
    }
}
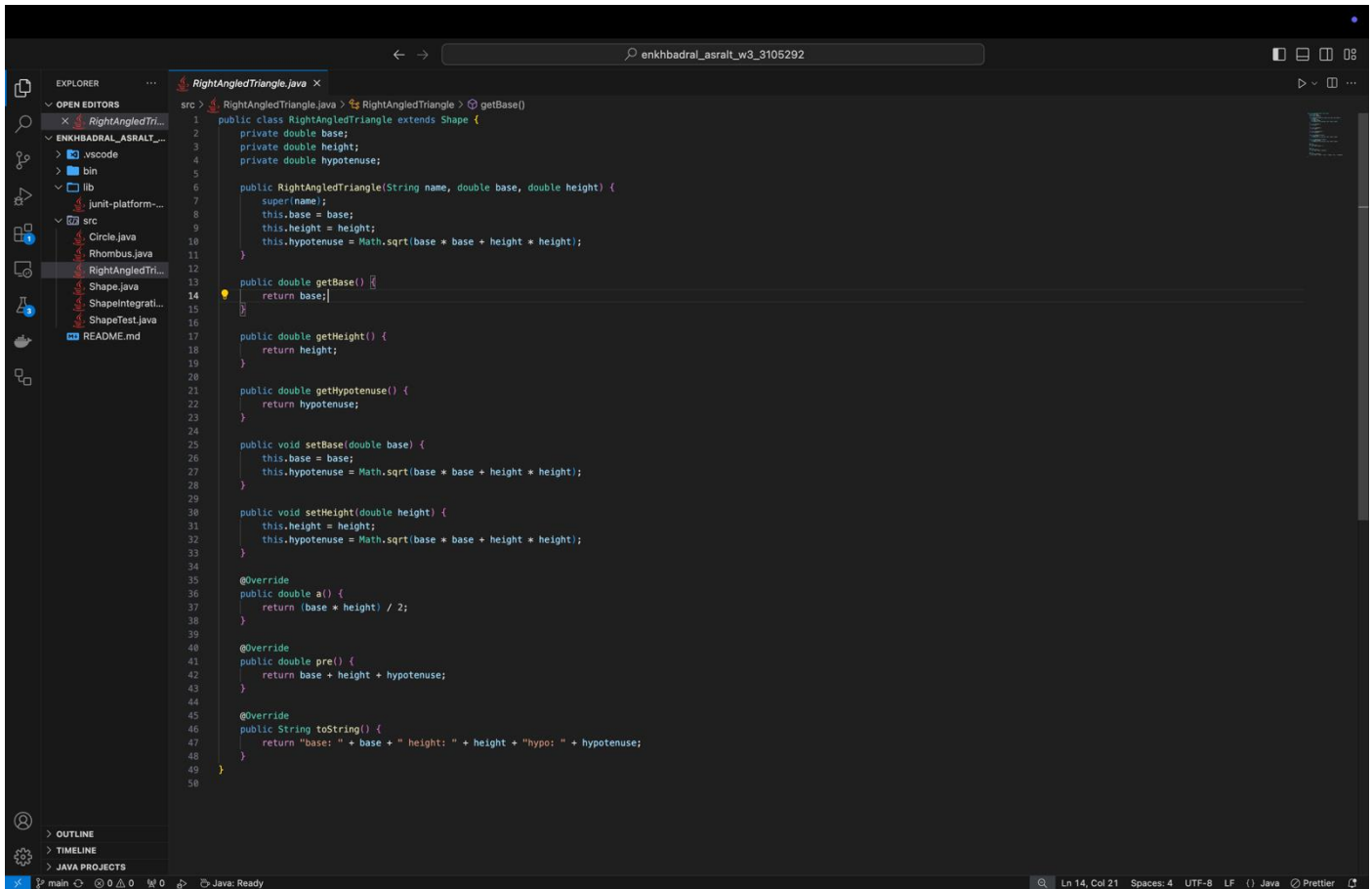```

```java
public class Rhombus extends Shape {

    public double getDiag1() {
        return diag1;
    }

    public double getDiag2() {
        return diag2;
    }

    public double getSide() {
        {
            return side;
        }
    }

    public void setDiag1(double diag1) {
        this.diag1 = diag1;
    }

    public void setDiag2(double diag2) {
        this.diag2 = diag2;
    }

    public void serSide(double side) {
        this.side = side;
    }

    @Override
    public double a() {
        return (diag1 * diag2) / 2;
    }

    @Override
    public double pre() {
        return side * 4;
    }

    @Override
    public String toString() {
        return "diagg" + diag1 + "diag2 " + diag2 + "sides " + side;
    }
}
```
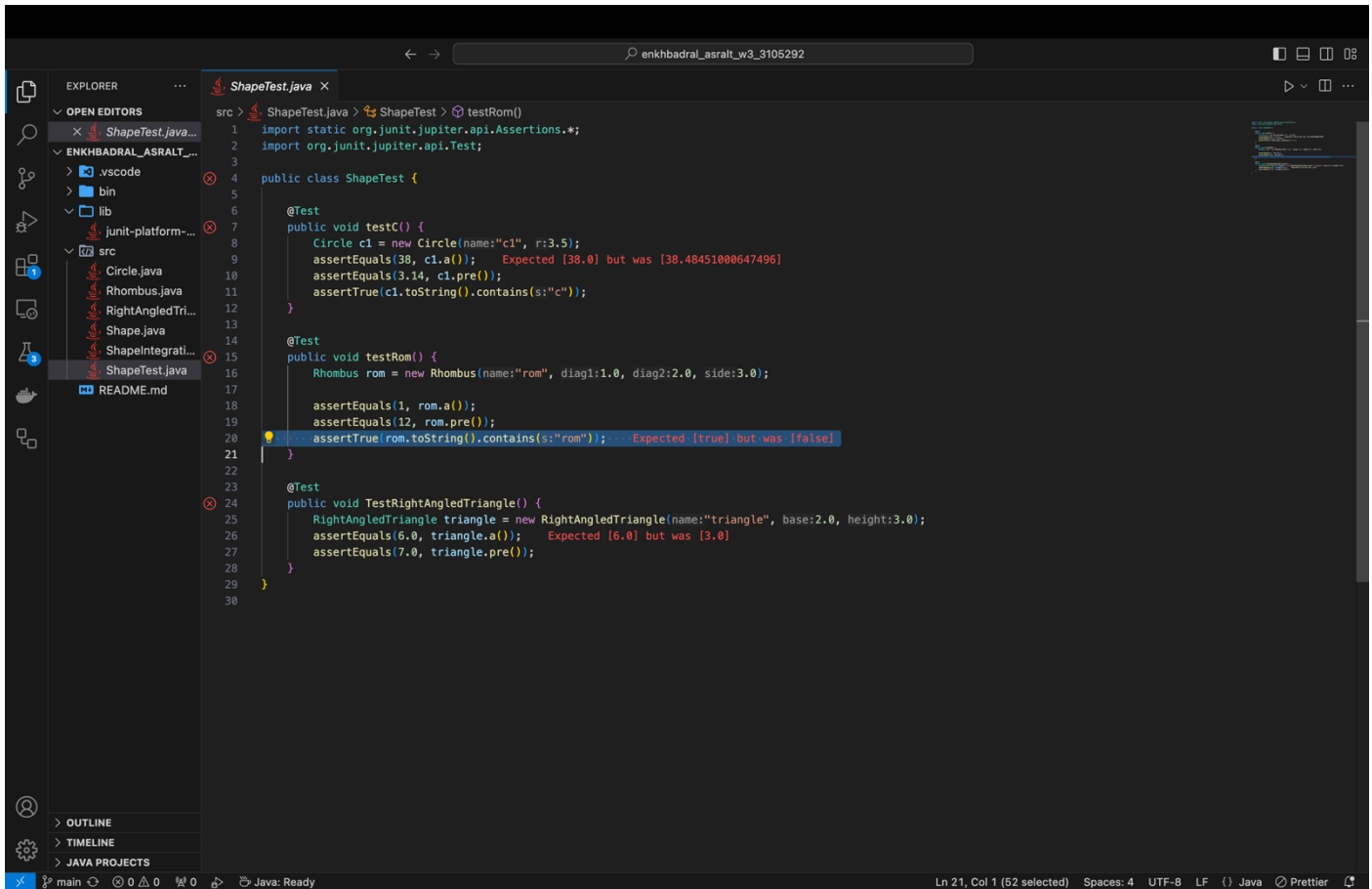
```java
public class RightAngledTriangle extends Shape {
    private double base;
    private double height;
    private double hypotenuse;

    public RightAngledTriangle(String name, double base, double height) {
        super(name);
        this.base = base;
        this.height = height;
        this.hypotenuse = Math.sqrt(base * base + height * height);
    }

    public double getBase() {
        return base;
    }

    public double getHeight() {
        return height;
    }

    public double getHypotenuse() {
        return hypotenuse;
    }

    public void setBase(double base) {
        this.base = base;
        this.hypotenuse = Math.sqrt(base * base + height * height);
    }

    public void setHeight(double height) {
        this.height = height;
        this.hypotenuse = Math.sqrt(base * base + height * height);
    }

    @Override
    public double a() {
        return (base * height) / 2;
    }

    @Override
    public double pre() {
        return base + height + hypotenuse;
    }

    @Override
    public String toString() {
        return "base: " + base + " height: " + height + "hypo: " + hypotenuse;
    }
}
```
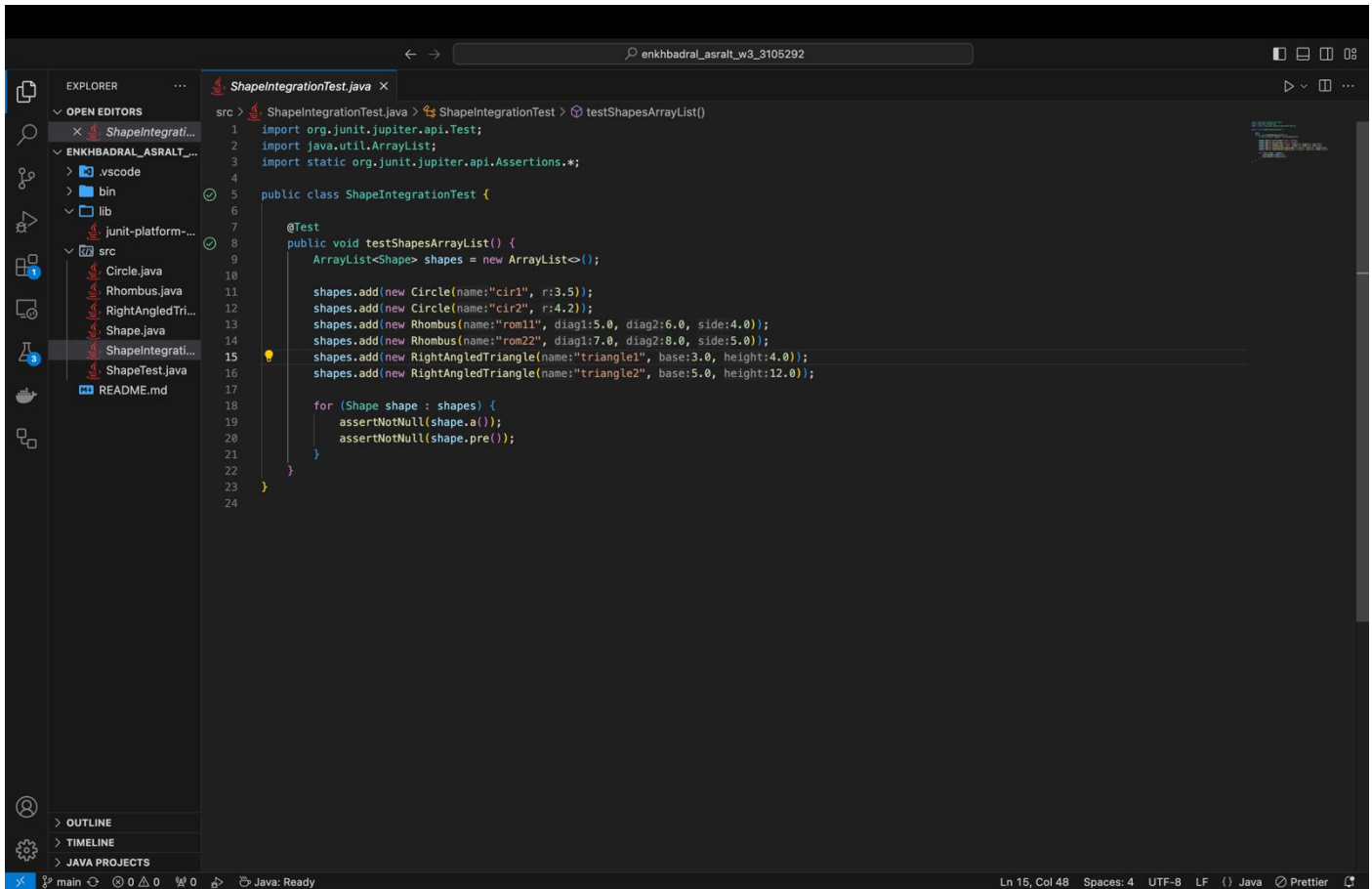
Task2.

```java
import static org.junit.jupiter.api.Assertions.*;
import org.junit.jupiter.api.Test;

public class ShapeTest {

    @Test
    public void testC() {
        Circle c1 = new Circle(name:"c1", r:3.5);
        assertEquals(38, c1.a());       Expected [38.0] but was [38.48451000647496]
        assertEquals(3.14, c1.pre());
        assertTrue(c1.toString().contains(s:"c"));
    }

    @Test
    public void testRom() {
        Rhombus rom = new Rhombus(name:"rom", diag1:1.0, diag2:2.0, side:3.0);

        assertEquals(1, rom.a());
        assertEquals(12, rom.pre());
        assertTrue(rom.toString().contains(s:"rom"));    Expected [true] but was [false]
    }

    @Test
    public void TestRightAngledTriangle() {
        RightAngledTriangle triangle = new RightAngledTriangle(name:"triangle", base:2.0, height:3.0);
        assertEquals(6.0, triangle.a());     Expected [6.0] but was [3.0]
        assertEquals(7.0, triangle.pre());
    }
}
```

Task3.

```java
import org.junit.jupiter.api.Test;
import java.util.ArrayList;
import static org.junit.jupiter.api.Assertions.*;

public class ShapeIntegrationTest {

    @Test
    public void testShapesArrayList() {
        ArrayList<Shape> shapes = new ArrayList<>();

        shapes.add(new Circle(name:"cir1", r:3.5));
        shapes.add(new Circle(name:"cir2", r:4.2));
        shapes.add(new Rhombus(name:"rom11", diag1:5.0, diag2:6.0, side:4.0));
        shapes.add(new Rhombus(name:"rom22", diag1:7.0, diag2:8.0, side:5.0));
        shapes.add(new RightAngledTriangle(name:"triangle1", base:3.0, height:4.0));
        shapes.add(new RightAngledTriangle(name:"triangle2", base:5.0, height:12.0));

        for (Shape shape : shapes) {
            assertNotNull(shape.a());
            assertNotNull(shape.pre());
        }
    }
}
```