HW2任务说明

任务说明

基本完成作业要求，控制台输出结果在在文末。

任务代码

主类代码

```
public class EX2 {
    public static void testCircle() {
        Circle c1 = new Circle();
        Circle c2 = new Circle(2);
        Circle c3 = new Circle(3, 4);
        Circle c4 = new Circle(5, 6, 7);
        System.out.println("now test Circle");
        c1.display();
        c2.display();
        c3.display();
        c4.display();
    }

    public static void testRectangle() {
        Rectangle r1 = new Rectangle();
        Rectangle r2 = new Rectangle(2);
        Rectangle r3 = new Rectangle(3, 4);
        Rectangle r4 = new Rectangle(5, 6, 7, 8);
        System.out.println("now test Rectangle");
        r1.display();
        r2.display();
        r3.display();
        r4.display();
    }

    public static void testTriangle() {
        Triangle t1 = new Triangle();
        Triangle t2 = new Triangle(3, 4);
        System.out.println("now test Triangle");
        t1.display();
        t2.display();
    }

    public static void main(String[] args) {
        testCircle();
        testRectangle();
        testTriangle();
        System.out.println("Total number of triangle: " + Triangle.printNum());
        System.out.println("Total number of shapes: " + Shape.printNum());

```

```
41
42      }
43  }
```

Shape类代码

```
1  //Shape 为抽象类，包含成员变量x，y为原点坐标
2
3  public abstract class Shape {
4      protected double x;
5      protected double y;
6      //静态变量num，用于存储实例化的对象个数
7      public static int num = 0;
8
9      public Shape() {
10         this.x = 0;
11         this.y = 0;//缺省值为0
12         num++;
13     }
14
15     public Shape(double x, double y) {
16         this.x = x;
17         this.y = y;
18         num++;
19     }
20
21     //静态方法，用于获取对象个数
22     public static int printNum() {
23         return num;
24     }
25
26     //抽象方法calArea，用于计算面积
27     public abstract double calArea();
28
29     //打印原点坐标的方法
30     public void display() {
31         System.out.print("Shape: ");
32         System.out.println("x=" + x + ",y=" + y);
33     }
34
35 }
```

IPerimeter接口代码

```java
//该为文件为一个接口，用于定义周长的方法
public interface IPerimeter {
    //常量PI
    public static final double PI = 3.14;
    //抽象方法，用于计算周长
    public abstract double calPerimeter();
}

```

Circle类代码

```java
//该类继承自Shape类，并实现IPerimeter接口
public class Circle extends Shape implements IPerimeter {
    //新增成员变量r
    protected double r;

    //构造方法
    public Circle() {
        super();
        this.r = 1;
    }

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

    public Circle(double x, double y) {
        super(x, y);
        this.r = 1;
    }

    public Circle(double x, double y, double r) {
        super(x, y);
        this.r = r;
    }
    //重写display方法，打印圆信息
    public void display() {
        System.out.print("Circle: ");
        System.out.println("x=" + x + ",y=" + y + ",r=" + r);
        System.out.println("Area=" + calArea() + ",Perimeter=" +
calPerimeter());
        System.out.println("**************************");
    }

    @Override
    public double calPerimeter() {
        return 2 * PI * r;
    }

    @Override
```

```
    public double calArea() {
        return PI * r * r;
    }
}

```

Rectangle类代码

```
//该类继承自Shape类，并实现IPerimeter接口
public class Rectangle extends Shape implements IPerimeter {
    //新增成员变量r
    protected double w;
    protected double h;

    //构造方法
    public Rectangle() {
        super();
        this.w = 1;
        this.h = 1;
    }

    public Rectangle(double l) {//此处l为对角线长度
        super();
        this.w = l/1.414;
        this.h = l/1.414;
    }

    public Rectangle(double w, double h) {
        super();
        this.w = w;
        this.h = h;
    }


    public Rectangle(double x, double y, double w, double h) {
        super(x, y);
        this.w = w;
        this.h = h;
    }
    //重写display方法，打印矩形信息
    public void display() {
        System.out.print("Rectangle: ");
        System.out.println("x=" + x + ",y=" + y + ",w=" + w + ",h=" + h);
        System.out.println("Area=" + calArea() + ",Perimeter=" +
calPerimeter());
        System.out.println("**************************");
    }

    @Override
    public double calPerimeter() {
        return 2 * (w + h);
```

```
    }

    @Override
    public double calArea() {
        return w * h;
    }
}

```

Triangle类代码

```
//该类继承自Shape类，并实现IPerimeter接口。新增成员变量d、h分别为三角形的底和高。
public class Triangle extends Shape implements IPerimeter{
    protected double d;
    protected double h;
    public static int tnum=0;

    public Triangle(){
        super();
        this.d=1;
        this.h=1;
        tnum++;
    }

    public Triangle(double d,double h){
        super();
        this.d=d;
        this.h=h;
        tnum++;
    }
    //隐藏父类的printNum方法
    public static int printNum(){
        return tnum;
    }

    //重写display方法，打印三角形信息
    public void display(){
        System.out.print("Triangle: ");
        System.out.println("x="+x+",y="+y+",d="+d+",h="+h);
        System.out.println("Area="+calArea());
        System.out.println("***************************");
    }

    @Override
    public double calArea() {
        return d*h/2;
    }

    @Override
    public double calPerimeter() {
        return 0;
```

```
    }
}

```

修改后的Triangle类calArea方法

```
    public double calArea() {
        try {
            if (d <= 0 || h <= 0) {
                throw new IllegalArgumentException("d or h is illegal!");
            }
        } catch (IllegalArgumentException e) {
            System.out.println(e.getMessage());
            return 0;
        }

        //若三角形两边之和小于第三边，抛入IllegalArgumentException异常
        double test1 = l1 + l2 - l3;
        double test2 = l1 + l3 - l2;
        double test3 = l2 + l3 - l1;
        try {
            if (test1 <= 0 || test2 <= 0 || test3 <= 0) {
                throw new IllegalArgumentException("The sum of two sides is less than the third side!");
            }
        } catch (IllegalArgumentException e) {
            System.out.println(e.getMessage());
            return 0;
        }
        return d * h / 2;
    }
```

主类testArrayList()函数

```
    public static void testArrayList() {
        //创建一个ArrayList对象listT，储存10个Triangle对象，参数由随机数生成
        ArrayList<Triangle> listT = new ArrayList<Triangle>();
        for (int i = 0; i < 10; i++) {
            listT.add(new Triangle(Math.random() * 10, Math.random() * 10));
        }
        test(listT);
    }
```

主类test()函数

```java
    public static void test(ArrayList<Triangle> listT) {
        //使用匿名内部类新建线程，实现上述ArrayList集合内所有triangle的面积计算，并打印至控制台。
        new Thread(() -> {
            for (int i = 0; i < 10; i++) {
                System.out.println("Area=" + listT.get(i).calArea());
            }
            System.out.println("**************************");
        }).start();
    }
```

修改后主类主函数

```java
    public static void main(String[] args) {
        System.out.println("now test illegal input");
        Triangle t3 = new Triangle(-1, 4);
        System.out.println("the result is " + t3.calArea());
        System.out.println("**************************");
        System.out.println("now test ArrayList");
        testArrayList();
    }
```

修改前控制台输出

```
now test Circle
Circle: x=0.0,y=0.0,r=1.0
Area=3.14,Perimeter=6.28
**************************
Circle: x=0.0,y=0.0,r=2.0
Area=12.56,Perimeter=12.56
**************************
Circle: x=3.0,y=4.0,r=1.0
Area=3.14,Perimeter=6.28
**************************
Circle: x=5.0,y=6.0,r=7.0
Area=153.86,Perimeter=43.96
**************************
now test Rectangle
Rectangle: x=0.0,y=0.0,w=1.0,h=1.0
Area=1.0,Perimeter=4.0
**************************
Rectangle: x=0.0,y=0.0,w=1.4144271570014144,h=1.4144271570014144
Area=2.000604182463104,Perimeter=5.657708628005658
**************************
Rectangle: x=0.0,y=0.0,w=3.0,h=4.0
Area=12.0,Perimeter=14.0
**************************
Rectangle: x=5.0,y=6.0,w=7.0,h=8.0
```

```
Area=56.0,Perimeter=30.0
**************************
now test Triangle
Triangle: x=0.0,y=0.0,d=1.0,h=1.0
Area=0.5
**************************
Triangle: x=0.0,y=0.0,d=3.0,h=4.0
Area=6.0
**************************
Total number of triangle: 2
Total number of shapes: 10
```

修改后控制台输出

```
now test illegal input
d or h is illegal!
the result is 0.0
**************************
now test ArrayList
Area=1.8342788203187799
Area=19.66526169017068
Area=35.7337242564128
Area=15.393427002943557
Area=0.04482577605783023
Area=6.660670070378015
Area=31.66326806998601
Area=17.261712464769914
Area=2.3326012067077513
Area=7.518907347818507
**************************
```