Java 语言程序设计作业 3

【实验要求】

- 1. 请严格按照所给的类名、函数名进行命名。函数需要严格按照给定的名字、参数、返回值定义和实现。严格区分大小写,不符合要求的命名视为错误。
- 2. 每个小题放置在不同的包中,包的命名为功能类的命名,包名需要小写。同学需要在包中实现对应功能类。每个类文件(.java)必须有 package 信息。
- 3. 作业文件夹请打包上传;上交的作业中需包含以学号命名的文件夹。

【测试类使用说明】

作业中的一些题会提供测试类。测试类会在每道题对应的包下,命名为 Test.java。测试类会调用同学们编写的功能类,同学们在编写完每一题的功能类后, 编译运行整个包,就可以得到功能类的运行结果。如果编译运行成功,那么说明同学 编写的功能类的接口是正确的。一些注意事项:

- 1. 测试类不需要同学们编写和修改。
- 2. 测试类可能会包含一些样例检查功能类是否编写正确。但是在作业批改中,会有 更多的样例测试功能类是否编写正确。
- 3. 提交的文件夹中可以保留或不保留测试类。
- 4. 测试类中已包含 main 函数,请勿再在功能类中实现 main 函数。

第一题:图形的多态

在包 shape 中实现基于多态的不同图形的功能类(详见下图)。

- (1) 创建抽象基类 Shape, 用来被其他各种其他抽象类继承。
 - a) 为类 Shape 创建两个构造函数:无参数;参数为一个字符串,表示这个 图形的颜色。
 - b) 为类 Shape 创建函数 getColor、setColor,功能为得到图形的颜色和填充新的颜色。getColor 无参数,返回值为一个字符串表示颜色;setColor 参数为一个字符串表示颜色,无返回值。
 - c) 为类 Shape 创建函数 isFilled, 功能为表示这个图形是否被着色, 无参数, 返回值为布尔类型。
 - d) 为类 Shape 创建抽象函数 getArea、getPerimeter, 功能为得到该图形的 面积和周长, 无参数, 返回值为一个实数。
 - e) 为类 Shape 创建函数或抽象函数 toString,无参数,返回值为一个字符串,格式为"图形名 (颜色)"。如果未着色,则颜色表示为"#"。
- (2) 创建类 Circle,表示圆形对象,继承自 Shape。
 - a) 为类 Circle 创建三个构造函数:无参数;参数为一个实数,表示半径; 参数为一个实数和一个字符串,表示半径和颜色。
 - b) 为类 Circle 创建函数 getRadius 和 setRadius,功能为得到圆的半径和设置新的半径。getRadius 无参数,返回值为一个实数表示半径; setRadius 参数为一个实数表示新的半径,无返回值。
 - c) 为类 Circle 重载 getArea、getPerimeter、实现圆形的面积和周长。
- (3) 创建类 Rectangle,表示矩形对象,继承自 Shape。
 - a) 为类 Rectangle 创建三个构造函数:无参数;参数为两个实数,表示宽和高;参数为两个实数和一个字符串,表示宽、高、颜色。
 - b) 为类 Rectangle 创建函数 getWidth、setWidth,功能为得到矩形的宽和设置新的宽;创建函数 getHeight、setHeight,功能为得到矩形的高和设置新的高。getWidth、getHeight 无参数,返回值为一个实数表示宽或高;setWidth、setHeight 参数为一个实数表示新的宽或高,无返回值。
 - c) 为类 Rectangle 重载 getArea、getPerimeter, 实现矩形的面积和周长。
- (4) 创建类 Square,表示正方形对象,继承自 Rectangle。
 - a) 为类 Square 创建三个构造函数:无参数;参数为一个实数,表示边长; 参数为一个实数和一个字符串,表示边长和颜色。
 - b) 为类 Square 创建函数 getSide 和 setSide,功能为得到正方形的边长和设置新的边长。getSide 无参数,返回值为一个实数表示边长;setSide 参数为一个实数表示新的半径,无返回值。
- (5) 创建类 Shapes,表示一系列图形的集合。
 - a) 为类 Shapes 创建一个构造函数,参数为一个 Shape[],表示初始化。
 - b) 为类 Shapes 创建函数 getArea、getFilledArea,表示所有图形集合的总面积、所有已着色图形的总面积。无参数,返回值为一个实数。
 - c) 为类 Shapes 创建函数 toString, 无参数, 返回值为一个字符串, 格式为 "[图形 1 (颜色 1), 图形 2 (颜色 2), ..., 图形 N (颜色 N)]"。

```
# color:String
                              # ...
                              + Shape()
                              + Shape(color:String)
                              + getColor():String
                              + setColor(color:String)
                              + isFilled():boolean
                              + getArea():double
                              + getPerimeter():double
                               + toString():String
               Circle
                                                                    Rectangle
                                             # width:double
# radius:double
                                              # height:double
+ Circle()
+ Circle(radius:double)
                                             + Rectangle()
+ Circle(radios:double,color:String)
                                             + Rectangle(width:double, height:double)
+ getRadius():double
                                             + Rectangle(width:double,height:double,color:String)
+ setRadius(radius:double)
                                             + getWidth():double
                                              + setWidth(width:double)
+ getArea():double
                                             + getHeight():double
+ getPerimeter():double
+ toString():String
                                              + setHeight(height:double)
                                             + getArea():double
                                             + getPerimeter():double
                                              + toString():String
                                                                      Square
                                                       # side:double
                                                       + Square()
                                                       + Square(side:double)
                Shapes
                                                       + Square(side:double,color:String)
     # shapeList:Shape[]
                                                       + getSide():double
     + Shapes(shapeList:Shape[])
                                                       + setSide(side:double)
     + getArea():double
                                                       + getArea():double
     + getFilledArea():double
                                                       + getPerimeter():double
     + toString():String
                                                       + toString():String
```

<abstract> Shape