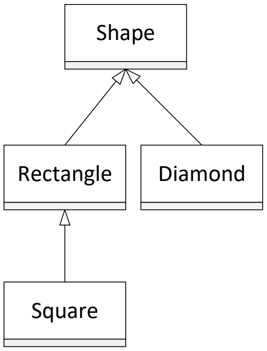
COMP2396 - Assignment 2

Due: 2 Mar, 2020 23:59

Introduction

This assignment tests your understanding of basic inheritance in Java.

You are asked to write a number of Java classes to model basic shapes: ***Shape***, ***Rectangle***, ***Square*** and ***Diamond***. ***Rectangle*** and ***Diamond*** are the subclasses of ***Shape***, and ***Square*** is the subclass of ***Rectangle***



You are also required to write **JavaDoc** for all non-private classes and non-private class members.

Implementation

The ***Shape*** class:

Methods

`Shape()`

Constructor, create an empty shape. A shape is typically a 2D array of pixels (boolean values).

`String toString()`

Return a drawing of the shape as a String. Each pixel should be represented by the character * and each empty space should be represented by the space character. Newlines are represented by newline character \n.

`int getArea()`

Return the area, i.e., the number of pixels in the shape.

`Shape intersect(Shape s)`

Return a new Shape object representing the intersection of this Shape object and s.

`Shape union(Shape s)`

Return a new Shape object representing the union of this Shape object and s.

The ***Rectangle*** class, a subclass of the ***Shape*** class:

Methods

`Rectangle(int width, int height)`

Constructor, create a rectangle with the specific `width` and `height`. For example, if `width` = 5 and `height` = 3, a drawing of this shape will look like this:

```
*****
*****
*****
```

The Diamond class, a subclass of the *Shape* class:

Methods

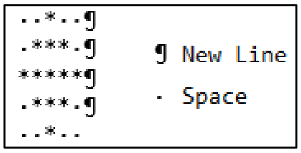
`Diamond(int size)`

Constructor, create a diamond shape with the specific size. For example, if `size` = 3, a drawing of this shape will look like this (the empty area is represented by space characters and each newline is represented by the newline character \n.):

```
  *
 ***
*****
 ***
  *
```

Return of toString():

Notice that there is no newline character after the drawing



```
..*..¶
.***.¶
*****¶
.***.¶
..*..
```

¶ New Line

· Space

The *Square* class, a subclass of the *Rectangle* class:

Methods

`Square(int size)`

Constructor, create a square with the specific `size`. For example, if `size` = 3, a drawing of this shape will look like this:

```
***
***
***
```

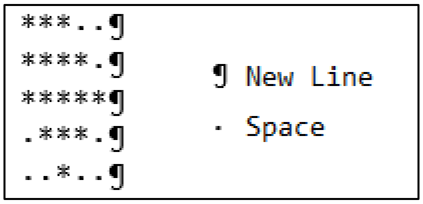
Grading

Your classes will be tested with a number of client programs. For example:

```
// ShapeTester.java
public class ShapeTester {
    public static void main(String [] args) {
        Square s = new Square(3);
        System.out.println(s);
        System.out.println("Area is " + s.getArea());
        Diamond d = new Diamond(3);
        System.out.println(d);
        System.out.println("Area is " + d.getArea());
        Shape union = s.union(d);
        System.out.println(union);
        System.out.println("Area is " + union.getArea());
        Shape intersect = s.intersect(d);
        System.out.println(intersect);
        System.out.println("Area is " + intersect.getArea());
    }
}
```

The following output is expected:

```
***
***
***
Area is 9
  *
 ***
*****
 ***
  *
Area is 13
***
****
*****
 ***
  *
Area is 16
  *
 **
***
Area is 6
```



```
***..¶
****.¶      ¶ New Line
*****¶
.***.¶      · Space
..*..¶
```

Notice that there is no empty line after the drawing.

- **80% marks** are given to the **correctness** of your program by the Moodle **evaluation** system.
  - ➢ You will see your marks immediately right after the evaluation.
  - ➢ You can re-submit the assignment before the deadline. We only consider the latest submission for final grading.
  - ➢ Output format is critical to the evaluation. Make sure there are space characters and newlines in proper locations in the program output.
  - ➢ You are not recommended to declare any static variable in your program as a static variable will keep data across test cases and make you difficult to debug.
  - ➢ Normally, we will not release any test case. However, if you failed to pass a certain test case, the expected output shown on the screen will give you some hints to debug the your program.
- **20% marks** are given to your **JavaDoc** and will be given **manually** after the due date. A complete JavaDoc includes documentation of every classes, member fields and methods that are not private.

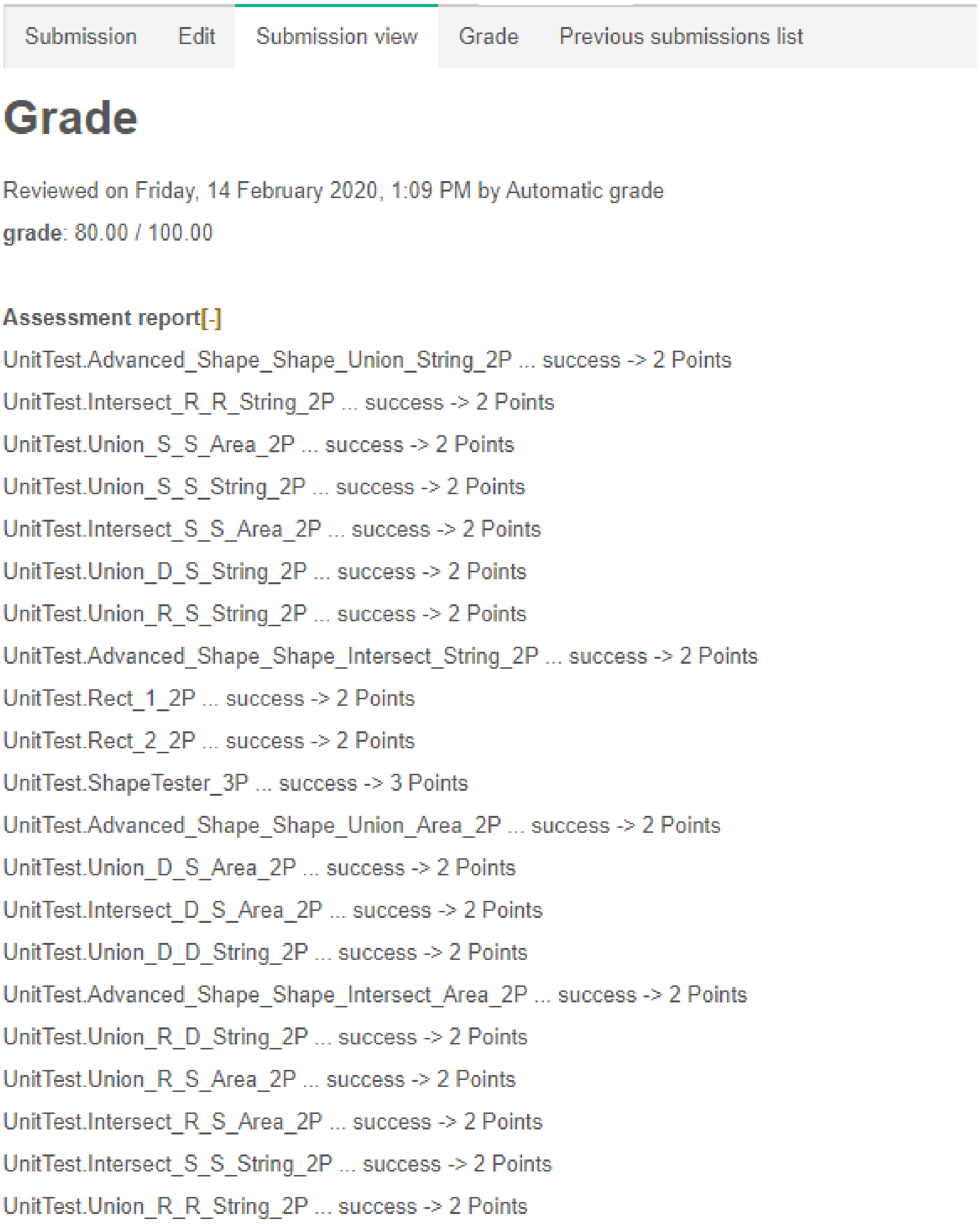
- **Economy is valuable in coding: the easiest way to ensure a bug-free line of code is not to write the line of code at all.**

Submission:

Please submit the following files to Moodle and evaluate. **Late submission is not allowed**.

- □ *Shape.java*
- □ *Rectangle.java*
- □ *Diamond.java*
- □ *Square.java*

If you got all the marks from the evaluation system (80% of the assignment), the grading report would look like this.



Grade

Reviewed on Friday, 14 February 2020, 1:09 PM by Automatic grade

grade: 80.00 / 100.00

Assessment report[-]

```
UnitTest.Advanced_Shape_Shape_Union_String_2P ... success -> 2 Points
UnitTest.Intersect_R_R_String_2P ... success -> 2 Points
UnitTest.Union_S_S_Area_2P ... success -> 2 Points
UnitTest.Union_S_S_String_2P ... success -> 2 Points
UnitTest.Intersect_S_S_Area_2P ... success -> 2 Points
UnitTest.Union_D_S_String_2P ... success -> 2 Points
UnitTest.Union_R_S_String_2P ... success -> 2 Points
UnitTest.Advanced_Shape_Shape_Intersect_String_2P ... success -> 2 Points
UnitTest.Rect_1_2P ... success -> 2 Points
UnitTest.Rect_2_2P ... success -> 2 Points
UnitTest.ShapeTester_3P ... success -> 3 Points
UnitTest.Advanced_Shape_Shape_Union_Area_2P ... success -> 2 Points
UnitTest.Union_D_S_Area_2P ... success -> 2 Points
UnitTest.Intersect_D_S_Area_2P ... success -> 2 Points
UnitTest.Union_D_D_String_2P ... success -> 2 Points
UnitTest.Advanced_Shape_Shape_Intersect_Area_2P ... success -> 2 Points
UnitTest.Union_R_D_String_2P ... success -> 2 Points
UnitTest.Union_R_S_Area_2P ... success -> 2 Points
UnitTest.Intersect_R_S_Area_2P ... success -> 2 Points
UnitTest.Intersect_S_S_String_2P ... success -> 2 Points
UnitTest.Union_R_R_String_2P ... success -> 2 Points
```