Programming Technology

Name: Murad Huseynov

Neptun ID: RX15MW

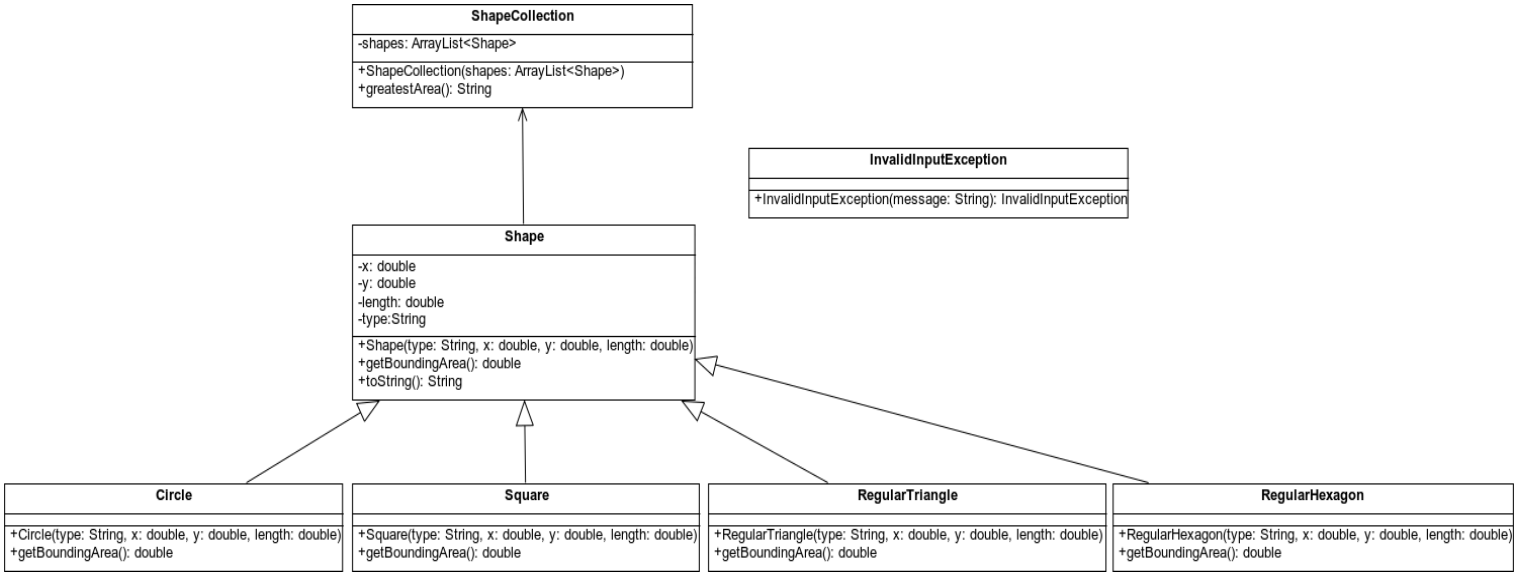
First Assignment: TASK 10.

Exercise Description

Fill a collection with several regular shapes (circle, regular triangle, square, regular hexagon). **Which shape has the greatest bounding box area?**

A bounding box of a shape covers the shape completely, and its sides are parallel with the x or y axis. Each shape can be represented by its center and side length (or radius), if we assume that one side of the polygons are parallel with x axis, and its nodes lies on or above this side. Load and create the shapes from a text file. The first line of the file contains the number of the shapes, and each following line contain a shape. The first character will identify the type of the shape, which is followed by the center coordinate and the side length or radius. Manage the shapes uniformly, so derive them from the same super class.

Class Diagram



Method Description

1) getBoundingArea(): double ➔ an abstract method declared in the abstract class 'Shape' used for calculating the bounding box area. For each shape, the method has been implemented in the relative class. The bounding box areas of each shape:
   - Circle = $4 * \text{length}^2$
   - Square = length^2
   - Regular Triangle = $(\text{length}^2) * \sqrt{3}/2$
   - Regular Hexagon = $(\text{length}^2) * \sqrt{3} * 2$

2) readFile(filename: String): ArrayList<Shape> ➔ this static method was used for reading text files in order to test the success/error cases.

3) greatestArea(): String ➔ this method finds and returns the type of the shape which has the greatest bounding box area.

4) toString(): String ➔ Overridden method in order to display the properties of shapes.

5) getType(): String ➔ returns the type of a shape.

6) getX(): double ➔ returns the 'x' coordinate of a shape.

7) getY(): double ➔ returns the 'y' coordinate of a shape.

8) getLength(): double ➔ returns the side length/radius of a shape.

Test Cases

1. Right input/ Valid test case ➔ input1.txt

2. Empty file/ error message: “File is empty or missing shape count on the first line” ➔ input2.txt

3. More than one integer on the first line/ error message: “Invalid format on the first line: expected single integer” ➔ input3.txt

4. Less number of shapes then the number that was specified in the first line/ error message: “Invalid input: Expected more shapes based on the first line count” ➔ input4.txt

5. The number of shapes exceeds the number that was specified in the first line/ error message: “Invalid input: More shapes in file than specified by the first line count” ➔ input5.txt

6. The test file is not formatted correctly/ error message: “Invalid input: Invalid format on line: T 2.3 1.4” ➔ input6.txt