KHQOK
Heng Freddy Zi Ping
ikhqok@inf.elte.hu

# Task

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.

# Analysis

Independent objects in the task are the shapes They can be divided into 4 different groups: circle, regular triangle, square, regular hexagon. All of them have the same formula to find the bounding box length and height.

The center coordinates for all shapes are denoted by x and y.

### Bounding Box

A bounding box is represented by four values:

- **(x_min, y_min)**: The bottom-left corner of the rectangle.
- **(x_max, y_max)**: The top-right corner of the rectangle.

The area of a rectangle is given by:

Area=length×height

Where:

- **length** = xmax−xmin
- **height** = ymax−ymin

**Square**

- **x_min =** x - halfLength
- **x_max** = x + halfLength
- **y_min =** y - halfLength
- **y_max** = y + halfLength

**Circle**

- **x_min =** x - rad
- **x_max** = x + rad
- **y_min =** y - rad
- **y_max** = y + rad

**Triangle**

We get the height by using the formula to get height for triangles. Sqrt(3)/2 * base. Where base = length.

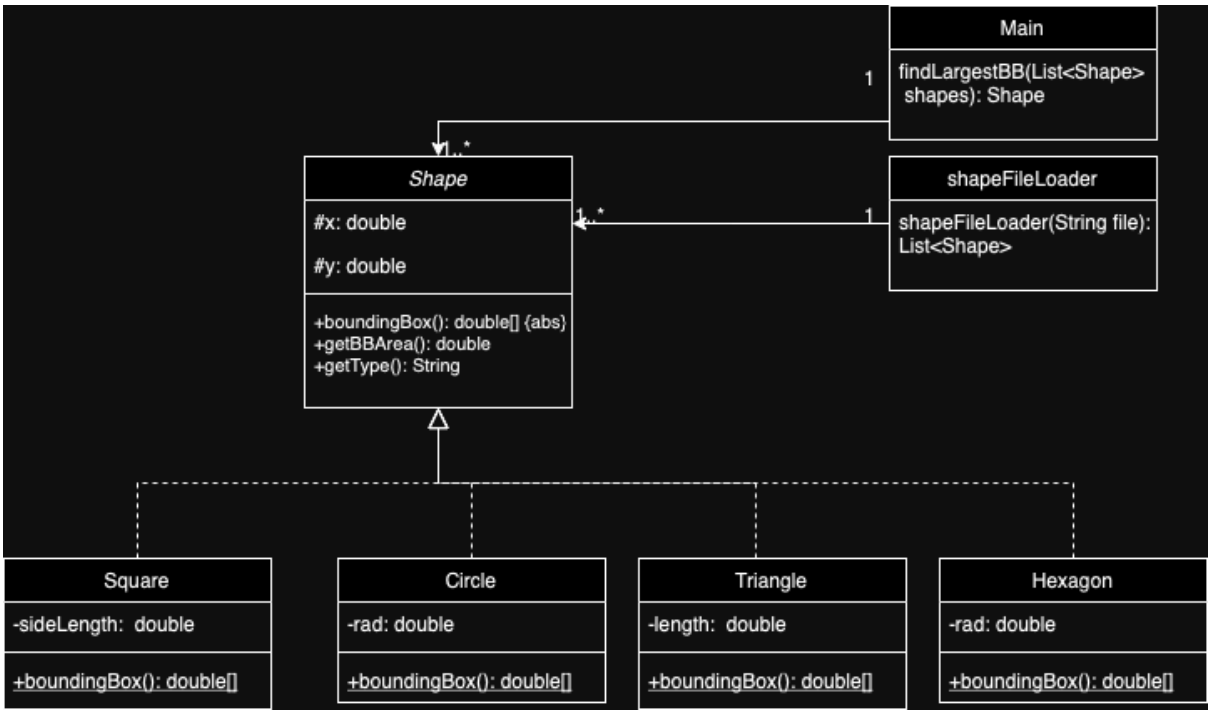
- **x_min =** x - height/2
- **x_max** = x + height/2
- **y_min =** y - height/2
- **y_max** = y + height/2

**Hexagon**

We get the height by using the formula to get height for hexagons. Sqrt(3) * radius. In a hexagon, the radius is equal to the length of each side,.

- **x_min =** x - radius
- **x_max** = x + radius
- **y_min =** y - height/2
- **y_max** = y + height/2

# UML Diagram




# Methods

| **Method** | **Description** |
|---|---|
| Constructor | Takes in 2 variables, x, y |
| getBBArea | Takes the third item of the array minus the first item of the array (x_max minus x_min) and takes the fourth item of the array minus the second item of the array (y_max mins y_min) |
| getType | Uses in built functions getClass and getSimpleName to get the shape of the name |
| ShapeFromFile | Uses BufferedReader to read each line of input, splitting it with “\\s+” for any white space, then by using switch cases, putting it into an array and creating a shape type based on the letter, center co-ords and length/radius provided by the input file |
| findLargestBB | Maximum search returning the largest shape. If this the shapes arrayList is empty, we return null, otherwise, we go through a |

| | |
|---|---|
| | foreach loop to check if any area is larger than the default value, set largestShape and largestArea to that shape and area, and return the largestShape |

# Whitebox Testing

| Test | Description | User Story | Expected Results |
|---|---|---|---|
| Negative Value Test (shapes_negCoord.txt) | The negative values should be converted to positive values. | As a user I want to be able to enter negative value given the center point is on the negative plane, when the program runs, then the negative values should work based on the logic | Largest Bounding Box: Square<br>The area is 2500.00 |
| Small Value Test (shapes_minVal.txt) | The tiny values (0.01) should work as it should as expected | As a user, I want to be able to enter values as small as possible when it is given as such, then the program should work no matter how small the numbers are | Largest Bounding Box: Circle<br>The area is 0.00 |
| Large Value Test (shapes_largeVal.txt) | Extremely large values should work as expected | As a user, I want to be able to enter values as large as possible when it is given as such, then the program should work no matter how large the numbers are | Largest Bounding Box: Hexagon<br>The area is 8660254037844387.00 |
| Single Shape Test (shapes_singleShape.txt)s | Should return the size and name of the shape | As a user, I want to be able to enter just one shape to get the area of the bounding box when I just want to calculate the area of a bounding box, | Largest Bounding Box: Square<br>The area is 2500.00 |

| | | | |
|---|---|---|---|
| | | then the program should return the area and name of the shape | |
| Identical Shape Test (shapes_identical Te | It will return the last shape in the array | As a user, I want to be able to enter just identical bounding boxes when the input file then the program should return the area and name of the last shape in the array | Largest Bounding Box: Square<br>The area is 100.00 |