First assignment

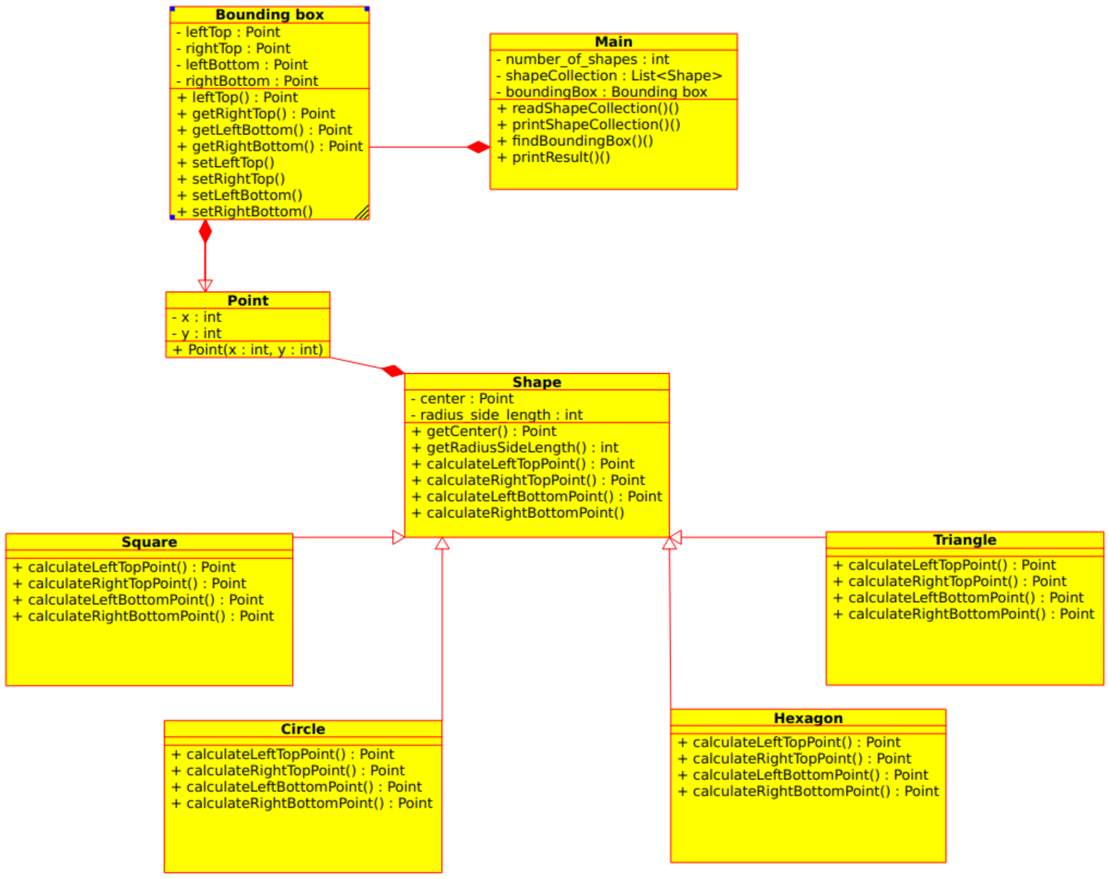
Student name: Khongorzul Khenchbish
Student id: r8hp78

Description:

7. Fill a collection with several regular shapes (circle, regular triangle, square, regular hexagon). **Determine the smallest bounding box, which contains all the shapes, and its sides parallel with an x or y axis.**

Each shape can be represented by its center and side length (or radius), if we assume that one sides of the polygons are parallel with the x axis, and its nodes lie 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 contains 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:


Bounding box
- leftTop : Point
- rightTop : Point
- leftBottom : Point
- rightBottom : Point
+ leftTop() : Point
+ getRightTop() : Point
+ getLeftBottom() : Point
+ getRightBottom() : Point
+ setLeftTop()
+ setRightTop()
+ setLeftBottom()
+ setRightBottom()
Main
- number_of_shapes : int
- shapeCollection : List<Shape>
- boundingBox : Bounding box
+ readShapeCollection()()
+ printShapeCollection()()
+ findBoundingBox()()
+ printResult()()
Point
- x : int
- y : int
+ Point(x : int, y : int)
Shape
- center : Point
- radius_side_length : int
+ getCenter() : Point
+ getRadiusSideLength() : int
+ calculateLeftTopPoint() : Point
+ calculateRightTopPoint() : Point
+ calculateLeftBottomPoint() : Point
+ calculateRightBottomPoint()
Square
+ calculateLeftTopPoint() : Point
+ calculateRightTopPoint() : Point
+ calculateLeftBottomPoint() : Point
+ calculateRightBottomPoint() : Point
Triangle
+ calculateLeftTopPoint() : Point
+ calculateRightTopPoint() : Point
+ calculateLeftBottomPoint() : Point
+ calculateRightBottomPoint() : Point
Circle
+ calculateLeftTopPoint() : Point
+ calculateRightTopPoint() : Point
+ calculateLeftBottomPoint() : Point
+ calculateRightBottomPoint() : Point
Hexagon
+ calculateLeftTopPoint() : Point
+ calculateRightTopPoint() : Point
+ calculateLeftBottomPoint() : Point
+ calculateRightBottomPoint() : Point

Short description of each methods:

Main class:

1. ***readShapeCollection()***
   It first reads the size of the collection then the inputs line by line and adds the shapes into a collection of type Shape from the input file. It throws an exception if the file is not found or empty. Summation algorithm is used here.

2. ***findBoundingBox()***
   Iterates through the collection of shapes created above and for each shape it checks whether the corners contain the bounding box's corresponding corner and if so it updates the bounding box's corner point, if not it does not update anything. It uses the maximum search algorithm pattern.

3. ***printResult()***
   Prints the corners of the bounding box.

Shape class:
Getters:

1. ***getRadiusSideLength()***
   Returns either radius or side length depending on the shape.

2. ***getCenter()***
   Returns the center coordinates as a Point object.

Calculation of corners:

3. ***calcLeftTop()***
   Returns the left top corner coordinate of the given shape. The calculation is overridden in the child classes depending on the shape type.

4. ***calcRightTop()***
   Returns the right top corner coordinate of the given shape. The calculation is overridden in the child classes depending on the shape type.

5. ***calcLeftBottom()***
   Returns the left bottom corner coordinate of the given shape. The calculation is overridden in the child classes depending on the shape type.

6. ***calcRightBottom()***
   Returns the right bottom corner coordinate of the given shape. The calculation is overridden in the child classes depending on the shape type.

Square, Circle, Hexagon, Triangle are the child classes inherited from parent class Shape.

Bounding box:
getters:

1. ***getLeftTop()***
   Returns the left top corner coordinate of the bounding box.

2. ***getRightTop()***
   Returns the right top corner coordinate of the bounding box.

3. ***getLeftBottom()***
   Returns the left bottom corner coordinate of the bounding box.

4. ***getRightBottom()***
   Returns the right bottom corner coordinate of the bounding box.

setters:

5. ***setLeftTop()***
   Sets the left top corner coordinate of the bounding box with the given coordinate.

6. ***setRightTop()***
   Sets the right top corner coordinate of the bounding box with the given coordinate.

7. ***setLeftBottom()***
   Sets the left bottom corner coordinate of the bounding box with the given coordinate.

8. ***setRightBottom()***
   Sets the right bottom corner coordinate of the bounding box with the given coordinate.

The testing (white box / black box):

1. ***empty-file.txt***
   used to check if the exception is triggered when the input is empty, because in this case, the collection is empty and we can not find the bounding box.

2. ***input1.txt***
   an input file where the shapes are not overlapped.

3. ***input2.txt***
   an input file where the shapes are overlapped, square shape contains other shapes inside so that the bounding box corners should be the same as square's corner coordinates.

4. ***input3.txt***
   In an input file where the shapes are overlapped, the circle shape contains other shapes inside so that the bounding box corners should be the same as the circle's corner coordinates.

5. ***input4.txt***
   In an input file where the shapes are overlapped, the triangle shape contains other shapes inside so that the bounding box corners should be the same as the triangle's corner coordinates.

6. ***input5.txt***
   In an input file where the shapes are overlapped, the hexagon shape contains other shapes inside so that the bounding box corners should be the same as the hexagon's corner coordinates.