Programming Technology

Assignment-1

Documentation for Task-6

Written and implemented by

Zayar Htet (DRCVG2)

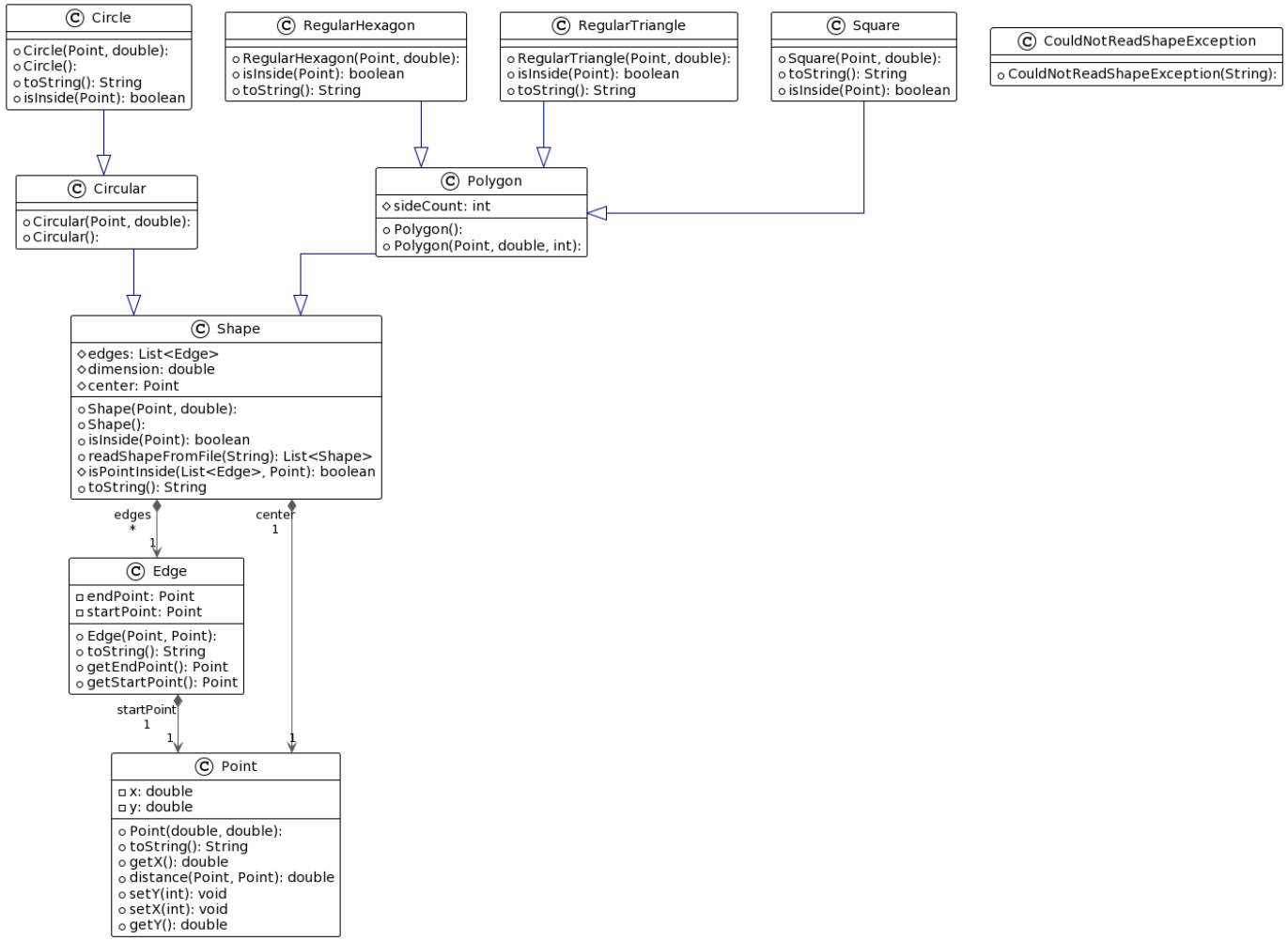
Task Description

Choose a point on the plane, and fill a collection with several regular shapes (circle, regular triangle, square, regular hexagon). How many shapes contain the given point?

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.

Solution

UML Diagram



Classes and Methods

Shape

1. ***isPointInside(edges: List<Edge>, target: Point) : Boolean*** – PROTECTED STATIC
   This is the static method of Shape which takes the list of polygon edges and a point as a parameter and then return the Boolean value if a point exists inside that given polygon. As this method is not related to the state of the object, so it is written with STATIC.
2. ***readShapeFromFile(filename: String) : List<Shape>*** - PUBLIC STATIC
   This method will take the string which is filename and will read the file and parse into the list of shape. As this method is not related to the state of the object, so it is written with STATIC.
3. **isInside(p: Point) : Boolean** - PUBLIC ABSTRACT NON-STATIC
   The children of this parent class will implement this method because there are 2 variants of shape, polygon and circular objects, therefore, this will be overridden in child classes.
4. **toString() : String** – PUBLIC ABTRACT
   The toString() method has to be implemented in child classes due to variants of shape.

Polygon, Square, RegularHexagon, RegularTriangle

1. **isInside(p: Point) : Boolean**
   This is the implementation of the isInside() method from parent class, Shape. This method will call the protected static method called isPointInside() from Shape class and return the Boolean value if a point exists inside that given polygon.
2. **toString() : String**
   String representation of the object.

Circular, Circle

1. **isInside(p: Point) : Boolean**

   This method will check if the radius is greater than the distance between the centre and the given point, which means this method will return the Boolean value if the point exists inside the circle.

2. **toString() : String**
   String representation of the circle.

Point

1. ***distance(p1,p2: Point) : double*** – STATIC
   This will return the double value of the distance between 2 points.

Edge - Collection of 2 Point object representing edges of the polygon.

Testing

There will be many input files to test the program according to the scenarios. The test cases cover all the possible scenarios.

Testing the business logic

- the point is inside the polygons
- the point is outside the polygons
- the point is on the edge of the polygons

Testing the counting algorithm

- there is no shape
- all the shapes do not contain the point
- all the shapes contain the point
- only the first shape contains the point
- only the last shape contains the point

Testing the exception, empty file and zero-shape file

FirstQuadrant, SecondQuadrant, ThirdQuadrant, FourthQuadrant

There are 2 parts in each test file, first part contains all the shapes that has the point inside, and the second part is the shapes that doesn't have the given point inside.

OnEdges - This test file will check if the program still works well if the point is existed on the edges of the polygon. In this test case, only the first quadrant will be used.

EmptyShape - This is the empty file

AllShape - All the shape contains the point

NoneShape - None of the shape contains the point

FirstShape - Only the firstShape contains the point

LastShape - Only the last shape contains the point

NullShape – Totally nothing in the file to test if the program handle this.

ExceptionShape – Test the exceptions inside the program