Gebze Technical University
Department of Computer Engineering
CSE 241/505
Object Oriented Programming
Fall 2018
Homework # 5
Inheritance

Due date Dec 23rd 2018

In this homework, you will write a Shape hierarchy for all the shapes that we defined in our previous homework assignments.

Class **Shape** is an abstract class that defines at least the following functions.

- **area** that returns the area of the shape
- **perimeter** that returns the perimeter
- Overloaded ++ and - - operators (both pre and post) for incrementing and decrementing the shape positions by 1.0.
- Overloaded operators == , !=, and other comparison operators to compare two shapes with respect to their areas.

Overloaded global operator<< can be used for printing the shapes to SVG files.

Class **Rectangle, Triangle, Circle** and **ComposedShape** are derived from base Shape class. They behave like the classes in HW3. Class ComposedShape keeps a vector of Shape pointers for the shape elements.

Polygon is an abstract class that derives from class Shape. **PolygonVect and PolygonDyn** are two concrete classes that derive from **Polygon** class. One of them uses STL vectors to keep the 2D points, the other uses dynamic memory to keep the 2D points.

We also define the following global functions

- Function printAll takes a vector of Shape pointers and prints all shapes to an SVG file
- Function printPoly takes a vector of Shape pointers and prints all Polygon shapes to an SVG file
- Function convertAll takes a vector of Shape pointers, converts all shapes to Polygons and returns a new vector with the new shapes.
- Function sortShapes takes a vector of Shape pointers and increasingly sorts them with respect to their areas.

Use the program that our TA Ahmet Soyyiğit showed you during the PS to draw this shape hierarchy In UML diagrams and submit the results.

Notes:

- Define and use your namespace.

- Do error and range checking for any parameters. Throw exceptions and test them in your client code. Do not forget to define the throw lists for your functions.
- For each class you will write its own header .h file and .cpp file for the separation of interface and implementation.
- As expected, you should follow all object-oriented programming principles.
- You should submit your work to the moodle page.
- You should submit the SVG files, the header and source code files and sample output results.