San José State University
Computer Science Department
CS151, Object Oriented Design and Programming, 05, Fall 2022

Homework #5

Objective:

This homework's objective is to review and understand the unit on the multithreading and concurrent programming lecture module.

Details:

Exercise 1:
In this exercise you will implement a program that computes area of multiple shapes in a concurrent manner.

Design and implement a class hierarchy to represent the following 2-D shapes: triangle, circle, rectangle, and hexagon. Remember that all these shapes are descendants from a basic shape and share some attributes and functionality. Each class needs to have getters and setters defined and implemented for all the attributes. Each leaf shape class (triangle, circle, rectangle, hexagon) should have *computeArea*() method implemented. Each leaf class should have at least one non-default constructor implemented to allow instantiating an object with appropriate attributes. For example, length and width attributes uniquely define a rectangle object. Each class should have *toString()* method implemented that outputs the values of all the attributes of the instance of that class to command line.

Define and implement class **Shapes**. This class should have a private *ArrayList* attribute that can contain shapes (any shape). Name this attribute shape*List*. Add the ability to add and remove objects from this attribute using methods *add()* and *remove()*. This class should have at least one constructor implemented. This class should also implement a getter and a setter for *shapeList*. Implement a synchronized public method called *compute()* in which you iterate through the members of *shapeList* and make a call to compute the shape's area and print the result to command line output.

Define and implement class **ShapeTest**. This class should implement *main()*. In the body of the *main()* method instantiate an object of *Shapes*. Then, instantiate at least 2 objects of each shape type (at least 8 shapes) and add to *Shapes*. The specific shape initialization parameters, such as length or width, are up to you. Make a call to *compute()* method in class Shapes. Make sure this method is called in multiple threads. This means that you will need to adjust your implementation of the above classes to allow for concurrent execution ability. Whether you want to use inheritance from a class *Thread* or implement *Runnable* interface is up to you. Save this class and its definition into a file named **ShapeTest.java**.

It is strongly suggested that in addition to **ShapeTest.java** file above you submit the following files: **Shape.java**, **Triangle.java**, **Circle.java**, **Rectangle.java**, **Hexagon.java, Shapes.java**.

Exercise 2:
Using the implementation of the shape hierarchy above, change this program to add *max()* and *min()* methods to class *Shapes*. These methods will return the shape with biggest or the smallest area respectively. In the *main()* method of **ShapeTest** class make a call to both max() and min() methods using the list you already initialized in the previous exercise. Print each of the returned object's details using a call to *toString()* method.

Submission:

Submit all files created by you for the homework exercises: Shape.java, Triangle.java, Circle.java, Rectangle.java, Hexagon.java, Shapes.java, ShapeTest.java and any other files you completed for this assignment, if any.

Make sure to submit by 11:59pm on the due date listed in Canvas. Submit your solution via Canvas.

If you have any questions, message me or the grader or both:
Yulia.Newton@sjsu.edu
madhujitaranjit.ambaskar@sjsu.edu

Grading:

Your code must compile and execute successfully in order to get full credit for this assignment. I will compile and execute the files.

- Program with no compile errors
- Program executes
- Program outputs what is required by the exercise

A total of 20 points are possible for this homework assignment.