| Name: Yixing Cao | Course name: EECS 3311 - Software Design | Section: Section B |
|---|---|---|
| Student ID: 216775140 | Project Name: "Lab 3 - project 1" | TA name: Alireza Naeiji |

**1.** This software project is all about creating an application that displays an interface with two buttons (Load shapes and Sort shapes). The two buttons respectively allow loading and sorting six shapes. Here are the two goals of this project:

(a) Each time we click on the **Load Shapes** button, it should instantiate six shapes (circles, rectangles, squares). It should also display the shapes on the interface.

(b) Each time we click on the **Sort Shapes** button, it should sort the six shapes based on their surface. Then, it displays the sorted shapes based on their sorting order. If the shapes are already sorted, it does nothing.

**2.** Challenges associated to the software project:

(a) Finding relevant/best resources for learning technologies and enhancing skills.

(b) Designing the roadmap of the project and implementing sorting algorithm for shapes.

(d) Testing the correctness and complete working of the project.

(e) Meeting the expectations and complete requirements of the project.

**3.** I will use the below concepts while working in this project:

(a) **Object/Class:** For representing each shape.

(b) **Inheritance:** In this project, inheritance will embody the generic concept of Shape (Shape is abstract class) so that all other classes like Circle, Square, and Rectangle can use it.

(c) **Encapsulation:** I will bundle Methods and Attributes related to a particular shape class in its class so that it could hide the internal representation/state of an object from the outside.

(d) **Class Diagram:** For describing the structure (attributes and methods of a particular class), and maintaining relationships between classes of the project/system.

(e) **Design Pattern:** It will help in analysing and designing the system with reusability and transparency. It will also help in clarifying the system architecture for building a better system/project solution.

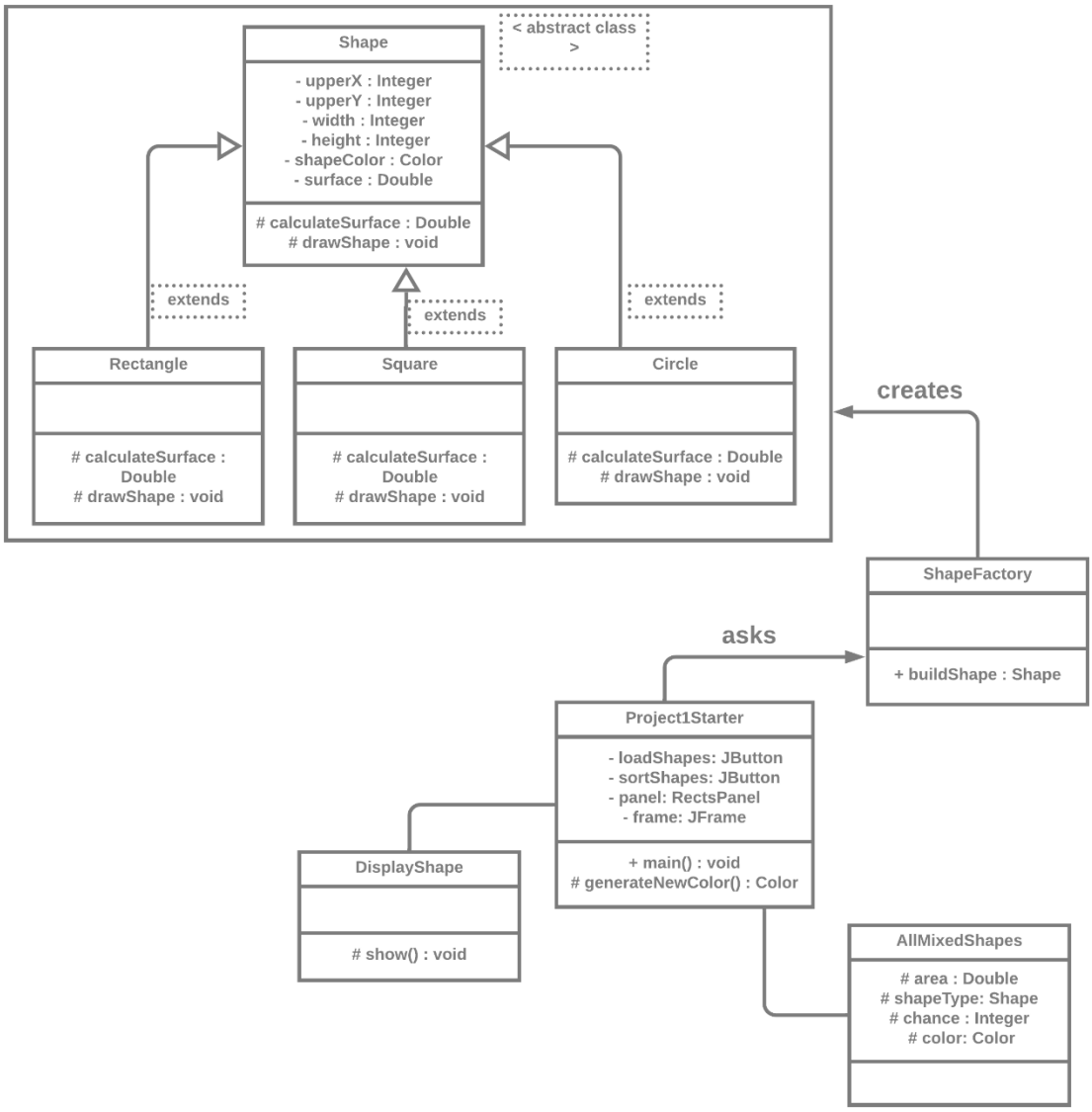
**4.** My reports will be structured in this way:

(a) Requirements Analysis

(b) Designing the system, (c) Implementation

# Part-II

1.




This above first UML class diagram is based on the Abstract Design Pattern Factory. In the above class diagram, I have used simple association between Project1Starter, DisplayShape, and AllMixedShape classes.

Then, I used Inheritance between Shape class and other three classes Rectangle, Circle, and Square. Then I combined all of these four classes in a container.

After that I used Realization/Implementation between this container and ShapeFactory class.

Again, ShapeFactory have realization/implementation relationship with Project1Starter class.

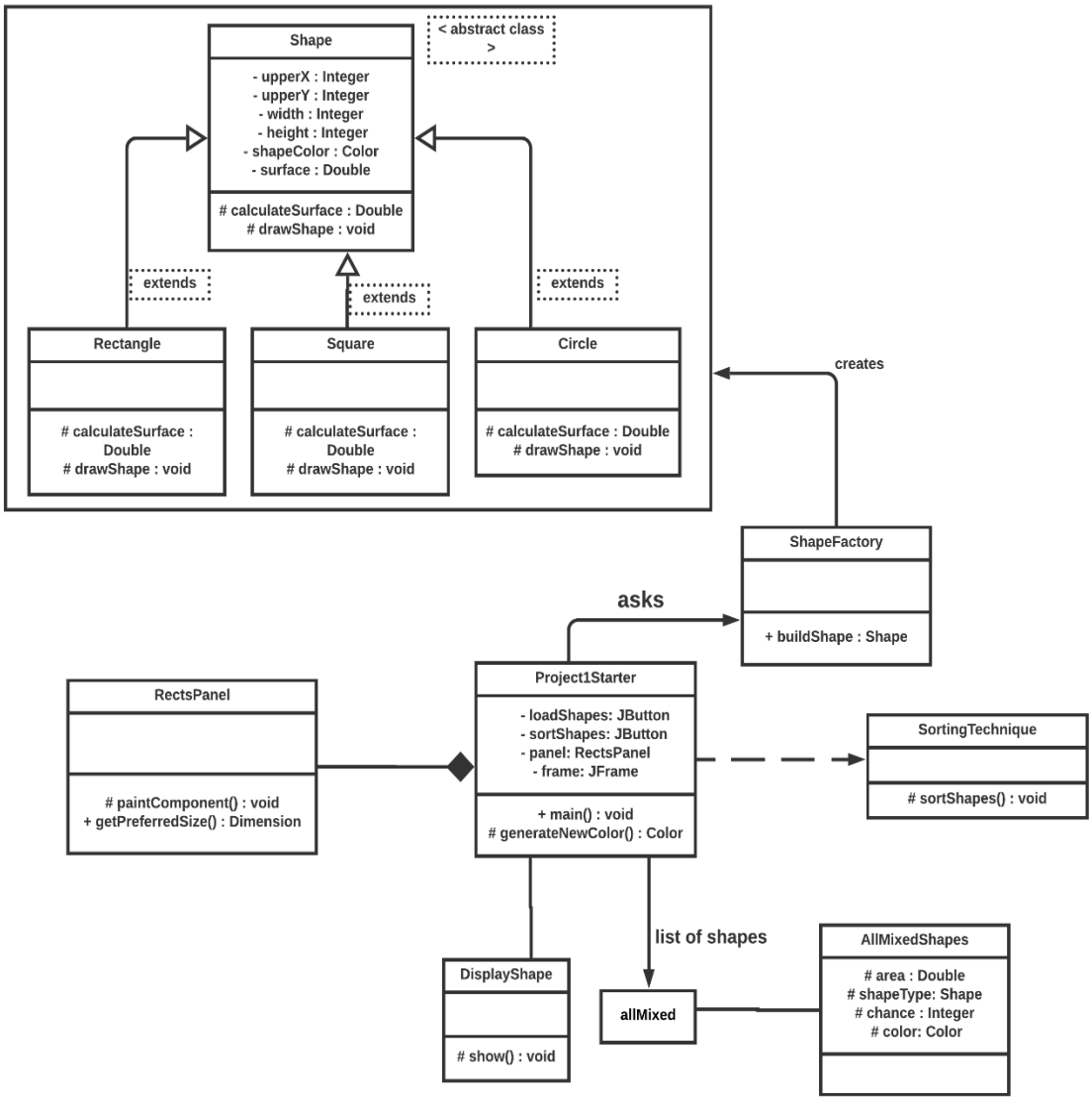
2. Below are the OO design principles which I have used in my class diagram:

(a) I have used Inheritance and Abstract class concept in the class diagram. Shape class will be an abstract class which contains the basic properties of any shape along with two abstract methods (calculateSurface() and drawShape()). Then, rest of the shape classes like Circle, Square, and Rectangle will extend this abstract class and implement both the abstract methods.

(b) I followed **Abstract Factory Design Pattern** while designing the above class diagram.

(c) In the above first UML class diagram, main(), generateNewColor(), show(), and buildShape() are the most relevant methods for the association relationship.

3.



In the second UML class diagram, all the things of the first UML class are already included. Apart from this, I have also included below more features in the second UML class diagram:

- I have included RectsPanel class, which is a private nested class in the Project1Starter class. It helps in painting the components and sets the preferred size of the Dimension.

- I have included **allMixed** list which is directly associated with **AllMixedShape** class because it contains the required properties for sorting the shapes.

- I have used composition relationship between **RectsPanel** and **Project1Starter**. So, once the Project1Starter object dies/destroys, Panel will be automatically destroyed because we really don't need any Panel if there is no project execution at all. That's why we can say that it follows good OO design principle.

Due to above these points, we can say that second UML class diagram yields a better design than the first UML class diagram.

## PART-III

1. I have used Bubble sort technique for sorting the shapes. Initially, I have created a class named “AllMixedShape” in which there is an attribute “area” for storing surface of the shapes. Also, this class has an extra variable "chance" which will help in identifying what type of shape is this. Then, I created a List of this class type which will store shapes along with their surfaces.

Then, I just performed Bubble sort (if the greater area shape is ahead in the list, then we need to swap it with lower surface shape so that sorting could be performed) on this list based on the surfaces. There is no need to use comparator here because we need to work with surface only. So, in this way, Bubble Sort will help in sorting shapes based on their surfaces.

2. As we can see in the UML class diagram, Initially, I created an abstract class named “Shape” which contains some properties/attributes for the shape body. Also, it has two protected abstract methods (calculateSurface() and drawShape()) which needs to be implemented by its child classes. calculateSurface() method in Shape class will help in calculating surfaces for the six different shapes in the JFrame. drawShape() will help in drawing the particular shape on the JFrame.

After that I created three classes Rectangle, Square, and Circle. These all three classes are derived class for the Shape class. These three classes extend Shape class. Also, these three classes will implement both the above abstract methods of the Shape class for calculating their own surfaces and for drawing its shape. There will be no other attributes/properties in these three classes because whatever is required for representing a shape is already present in the Shape class.

Now, in this UML class diagram, I used Abstract Factory Design Pattern for designing the system. So, I created another class named “ShapeFactory” which will instantiate all the six different shapes for the given different sizes once we call buildShape() method. This class will be used in “Project1Starter” which is the main class for starting the project execution because “Project1Starter” is the class which contains main() method for starting execution of the project.

Now, In the “Project1Starter” classes, I have added the GUI (Graphical User Interface) part. This class contains two JButtons, one JFrame, and one JPanel. One button “Load Shape” will instantiate six shapes (circles, rectangles, squares). It will also display the shapes on the interface. Another button “Sort Shape” will sort the six shapes based on their surface. And then it displays the sorted shapes based on their sorting order. If the shapes are already sorted, it does nothing.

There is another method generateNewColor() in Project1Starter class, which will generate new random RGB color combination every time we click on the “Load Shape” button.

There is one more private nested class “RectsPanel” implemented inside “Project1Stater” which will handle repaint() of the shapes along with preferred size of the dimension of the JPanel. This class helps in printing the components.
This class “Project1Starter” contains a list of “allMixed” Shapes which will store all the six different random shapes with their surfaces and color. This list of different all mixed shapes will be sorted based on their surfaces.

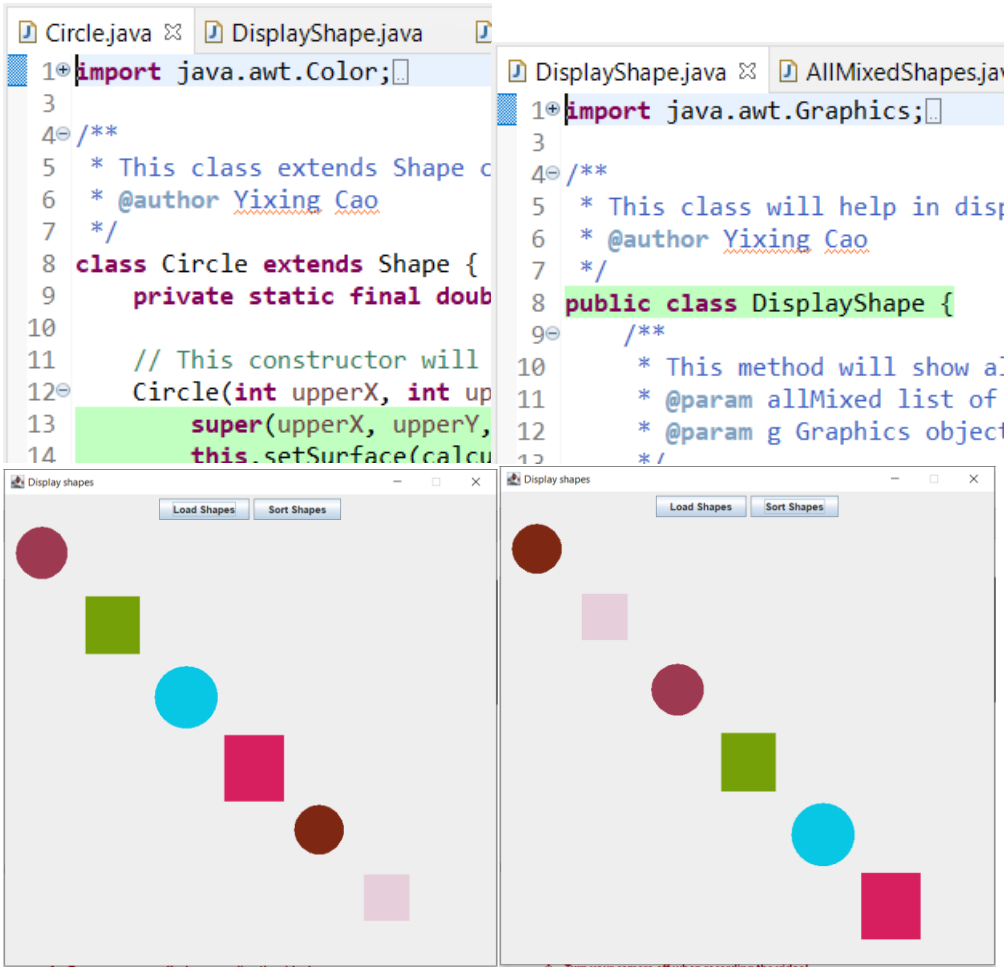
There is another class “SortingTechnique” which will perform sorting on the list of all mixed shapes. This sorting algorithm will be used once we click on the “Sort Shape” button. sortShapes() of “SortingTechnique” implements bubble sort algorithm for sorting list of all mixed shapes.

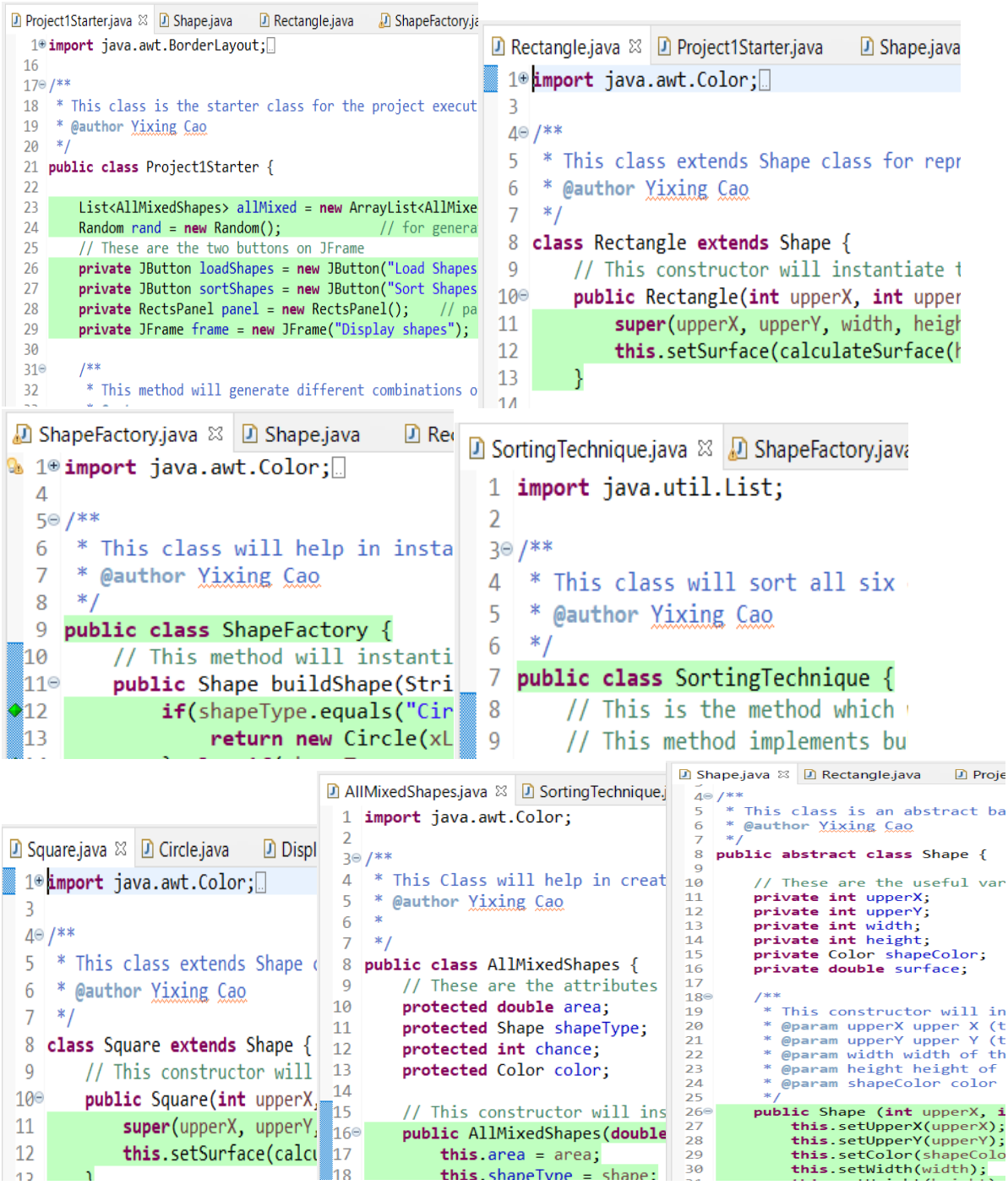
**Compile:** I used Eclipse IDE while working on the project. So, I used compile and run icon ( ) for compiling all the classes in the class diagram.

3. Eclipse IDE (Version: 2020-12 (4.18.0))

Java Version: JDK 14.0.2

4. Below are the snapshot of the execution of the code along with output of the application:






These above are all the images of the execution of the code which includes 9 classes (Square.java, Shape.java, AllMixedShape.java, SortingTechnique.java, Rectangle.java, Circle.java, Project1Starter.java, DisplayShape.java, and ShapeFactory.java) and two application window output for loading and sorting of the all different six shapes.

# Part-IV

1. Below are the things went well in this project:

(a) UML class diagram

(b) Implementation of shape classes like Shape, Rectangle, Square, and Circle classes.

(c) Creation of JFrame, JPanel, and JButtons.

(d) Implementing Sorting algorithm for sorting different shapes based on their surfaces.

2. Actually for the final delivery of project work, everything worked well. But, while implementing the project, I faced too much difficulties in implementing sorting algorithm for sorting the shapes based on their surfaces. My "Sort Shape" button was not working for so long, but after too much hard work and efforts, it started working. Also, setting the random shapes on the fixed coordinates on the JFrame took good amount of time and efforts.

3. I learnt about below things while implementing this project:

(a) JFrame, JButtons, JPanel.

(b) Got good confidence in implementing sorting algorithm for the different types of objects.

(c) Learnt about how to design the software project based on the given requirements.

(d) Learnt about how to draw UML class diagram for structuring the classes and maintaining the relationships between classes.

(e) Learnt about design patterns and some manual testing.

4. Below are my three recommendations to ease the completion of the software project:

(a) Understanding design pattern for designing UML class diagram so that designing of project could be easy and the relationships between classes could be identified easily. I found LucidChart is a great tool for creating such types of diagrams.

(b) Understand JFrame, JButtons with addActionListeners, and JPanel.

(c) Understand classes and objects in depth so that sorting algorithm could be implemented easily on the different types of objects.