Asanka Gunatilaka
Student ID: 217499450
Course: EECS3311
Section: B
Name of the software project: First Software Project
TA's name: Naeiji Alireza

Introduction (PART I)

Explain what the software project about and what are its goals:

Software project is about creating an interface that produce shapes, then having the ability to reorganize them in order of their size. The goals of this assignment are, being able to skillfully assign responsibilities to software objects and meet all the requirements of the assignment.

Explain the challenges associated to the software project:

Understanding design patterns and how to apply them into the design of my project. I also had problems with JSwing as this was the first time, I had used it. It was also challenging for me to ensure that OO design principles were included in my project.

Explain the concepts (e.g., OOD, OOD principles, design patterns) you will use to carry out the software project:

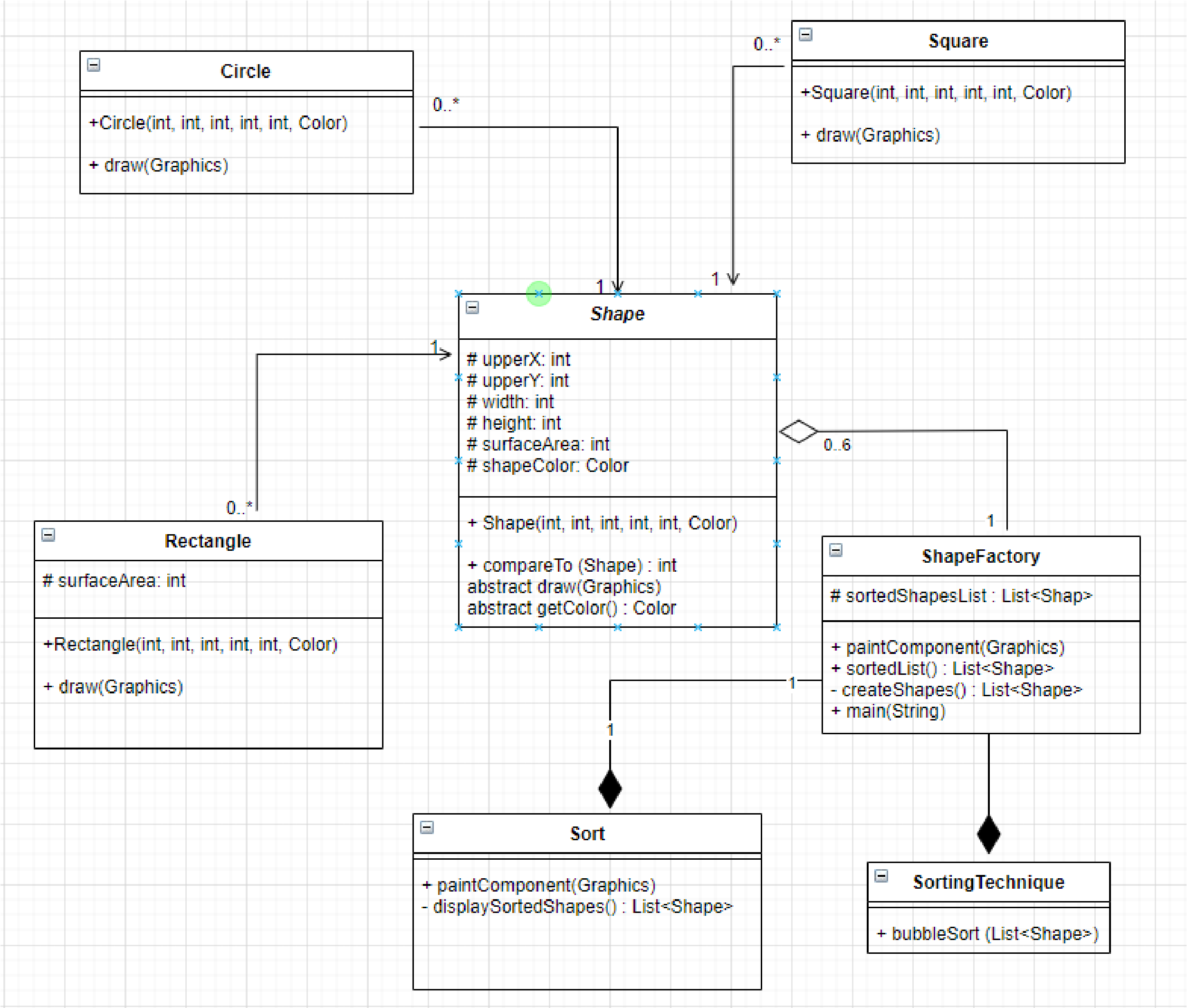
I used OO analysis to first define use cases and a domain model to help me with the interaction diagram and the UML diagrams. I have used all four of the design principles Abstraction, Encapsulation, Polymorphism, Inheritance in this project. In terms of design patterns I have used Singleton and Factory patterns.

Explain how you are going to structure you report accordingly:

I will be structuring the report based on the parts (part 1 – 4) that are required in this report. Each part will be listed, and description would be provided.

Part II: Design of the solution (PART II)

UML Diagram


Circle
+Circle(int, int, int, int, int, Color)
+ draw(Graphics)
0..*
0..*
Square
+Square(int, int, int, int, int, Color)
+ draw(Graphics)
1
1
Shape
# upperX: int
# upperY: int
# width: int
# height: int
# surfaceArea: int
# shapeColor: Color
+ Shape(int, int, int, int, int, Color)
+ compareTo (Shape) : int
abstract draw(Graphics)
abstract getColor() : Color
1
0..*
Rectangle
# surfaceArea: int
+Rectangle(int, int, int, int, int, Color)
+ draw(Graphics)
0..6
1
ShapeFactory
# sortedShapesList : List<Shap>
+ paintComponent(Graphics)
+ sortedList() : List<Shape>
- createShapes() : List<Shape>
+ main(String)
1
1
Sort
+ paintComponent(Graphics)
- displaySortedShapes() : List<Shape>
SortingTechnique
+ bubbleSort (List<Shape>)

Design principles used in the diagram:

I have used abstraction and inheritance in this project by using the ‘Shape’ class as an abstract class. This allowed me to create a base shape and have subclasses that define the actual shapes with custom features. Polymorphism was achieved by following this method.
Because I used an abstract class it allowed me to reuse code via inheritance as all shapes mostly have similar fields and functionality.
By making fields private/protected I have encapsulated data and only allowed access to fields via access modifiers.

Implementation of the solution (Part III)

Describe the algorithm of the sorting technique you have used to sort the shapes:

I used the bubble sort algorithm for this project due to its simplicity and the fact that not many elements are in added into the list we are sorting. It works by repeatedly swapping the adjacent elements if they are in wrong order until the entire list sorted. In order to compare two shapes I am using a custom compareTo method in the Shape class. I perform the bubble sort from the list that is received by the ShapeFactory and return back the sorted list to it can be displayed on the JFrame.

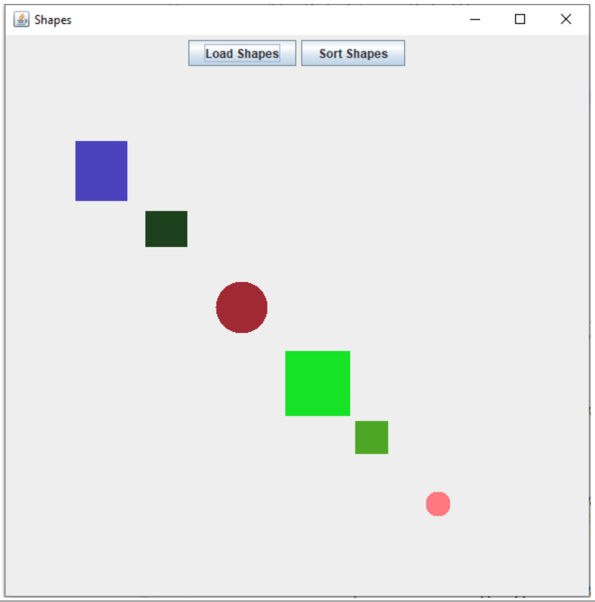
Describe how you have implemented and compiled all the classes of your class diagram in Java:

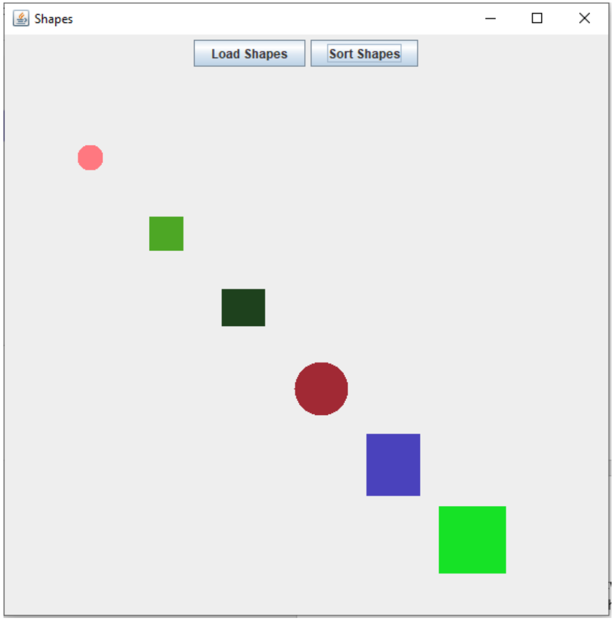
I implemented the project by following the two design patterns Singleton and Factory. I first, did an OO Analysis by defining use cases such as "User clicks on the button to create the shapes", "Start process of making shapes until 6 are done", "Pick a random shape", "Pick a random color", etc. Then did use cases for sorting the shapes. I then defined a domain model and decided to go with an abstract class as it could be used for inheritance. Then I did a rough sketch of an interaction diagram to get an idea of how the classes would interact. Then drew a diagram. However, I was not able to complete the UML diagram fully before I started the coding. I made some adjustments to the UML diagram as I coded.

Specify the tools you have used during the implementation: version of Eclipse/IntelliJ or of another IDE use to write code and run it, version of JDK, etc.:

I used Eclipse Version: 2020-09 (4.17.0), Java environment: JavaSE-1.8 | jdk-15.0.1

Take a snapshot of the execution of the code (i.e., of the interface) and comment it in the report:





Create a short video (2 to 3 mins) showing how to launch your application and run it:

See in Github.

Conclusion (PART IV)

What went well in the software project?

I followed the correct OO design workflow to get an idea of how to design the project. Doing this helped me put together foundation and without it I would have been lost as to where to even start. My prior knowledge of abstraction in java helped me relatively quickly code some of the parts.

What went wrong in the software project?

I spent a bit of time getting the button to trigger the display of shapes and the sorting of shapes. I also could have perhaps managed my time better for this project.

What have you learned from the software project?

OO Design plays a big part in a software project. It lays the foundation for the project and without it you would be lost as to where you are heading. I am used to starting coding right away when I receive an assignment and knowing now about the design workflow, I am able to plan a project properly.
Time management is also very important.

What are your top three recommendations to ease the completion of the software project?

1. Spend more time designing the project so I have a plan for what I need to code.
2. Plan out the time slots for the project so it's spread out evenly throughout the 3 weeks given for a project
3. Allocate extra time for software projects such as this as they are more time consuming