The Report for Display and Sort Shapes Application

EECS 3311 B

Misato Shimizu

215310246

Naeiji Alireza

**PART Ⅰ (Introduction)**

What is the software project about?

This software project creates 3 types of shapes: circle, rectangle and square. It consists of a load button, which creates 6 shapes randomly with random height, width and color, and a sort button which sorts those shapes based on the value of their surface area. The goal is to show 6 non-overlapping shapes with the load button, and to reorganize those shapes from left to right with smaller to bigger surfaces.

What are the challenges associated to the software project?

One of the challenges associated to this project is to create a complete report, which I have not experienced in previous computer science courses. Also, there are several unfamiliar tasks to complete such as UML class diagrams and design patterns.

What are the concepts I will use to carry out the software project?

I will use Factory pattern and Composition pattern for UML diagram, and inheritance in the main program. Factory pattern is one of creational patterns which consists of interface and does not show the process of creating shapes to clients. Composition pattern is also a creational pattern which has a component (Shape class) to manage other classes, leaves (Circle, Rectangle and Square class) and a composite (ShapeFactory class) to store shapes. Inheritance is used multiple times in the application to inherit methods and fields.

How am I going to structure my report accordingly?

My report consists of 4 major sections, Introduction to show the basic idea of this project, Design to show the organization of classes in code, Implementation to explain the process and to show the results and Conclusion to summarize my after thoughts.

**PART Ⅱ (Design)**

Diagram, schematic

Description automatically generated

This UML class diagram includes Factory pattern: There is an interface, Shapes, implemented by Circle, Square and Rectangle class, and these shapes can be created by ShapeFactory class. This method allows me to hide how shapes are created from other people. Moreover, the diagram includes Composite pattern as well: Shape class has a list of its own objects, and it can be a client and component. Circle, Rectangle and Square classes can be leaves, and ShapeFactory class can be a composite.

There is inheritance used in my class diagram: Shape class extends JPanel class so that it allows to show a canvas for drawing shapes, and SortingTechnique class extends ShapeFactory class so that it accesses all the surfaces of shapes for comparison. Moreover, the three concrete classes which are Circle, Rectangle and Square class extends ShapeFactory class. Each concrete class has an interface, which is Shapes class, including paintComponent() and getSurface() methods.

Diagram

Description automatically generated

This second design utilizes Decorator pattern with an interface Shape, implemented by Circle, Rectangle, Square and ShapeFactory classes. By extending ShapeFactory class in RedShape, YellowShape and BlueShape classes, we can create red, yellow, or blue colored shapes. However, this design is not better than the first one as it restricts types of colors for shapes. Moreover, using the first diagram with composite pattern is better because it groups circles, rectangles, and squares as shapes instead of dealing with them separately. As the purpose of using decorator pattern is mainly to add features to only one of objects and allow other objects not to be changed, it is useless for the current program which needs six random shapes with random colors.

**PART Ⅲ (Implementation)**

Regarding the algorithm of the sorting technique used to sort the shapes

The algorithm used to sort 6 shapes is bubble sorting. It first creates an ArrayList of 6 shapes and compare its surfaces. Using for loops, it will check a surface of a shape at position starting with 0 and compare it to the next shape in the list and swap these shapes if the first shape’s surface is smaller than the other. The loop continues until its index reaches the last position of the list.

Regarding how I have implemented and compiled all the classes of my class diagram in Java

I have implemented the first UML class diagram. First, we have to extend JPanel and implement ActionListener in Shape class to draw shapes on a panel and to add behaviors for buttons respectively. By instantiating an object of Shape and calling setBtn() method, it generates JFrame, two buttons and the range of the frame. When a load button is clicked, Shape class asks ShapeFactory class to create a random shape 6 times. In the ShapeFactory class, 3 shape classes (Circle, Rectangle and Square) are called to create their shapes with random width, height and color using an interface, Shapes. These shapes will be stored in a list in Shape class. Then, 6 random shapes with random colors will appear on the panel. When a sort button is clicked, remove all the shapes on the panel and set up an empty panel by calling setBtn() method. After setting it up, we create an object of SortingTechnique class, pass the list of Shapes and use its method, sort() to organize the six shapes based on their surface using Bubble Sort. As we have an organized Shape list, Shape class will draw each shape from a position 0 in the list to position 5. Clicking the sort button again does not generate any actions as the shapes are already sorted.

Regarding the tools I have used during the implementation

Eclipse 4.18.0 with JDK 15.0.1 is used for this project.

Snapshot of the execution of the code and comment it in the report

After running the program:

Graphical user interface, application, Word

Description automatically generated

When a load button is clicked:

Chart, waterfall chart

Description automatically generated

When a sort button is clicked:

Chart

Description automatically generated

A short video showing how to launch my application and run it

This video is included in the folder in GitHub.

**PART Ⅳ (Conclusion)**

What went well in the software project?

Sorting part of the program went smoothly as the amount of element was not large, and the used sorting technique, Bubble Sort, is simple.

What went wrong in the software project?

After completing the program, I realized that I could structure the whole project better in the design phase so that there are a smaller number of fields in Shape class. It would be better if I spent more time figuring out more efficient design structure.

What have I learned from the software project?

I have learned how to use JPanel and its components such as JButton and addActionListener. Also, one of major realization from completing this project is the importance of planning and designing the whole project beforehand. Although I spent most of time coding and fixing bugs when I created programs in previous computer science courses, this project let me consider of taking more time next time on planning for more efficiency.

What are my top 3 recommendations to ease the completion of the software project?

The first recommendation is to take adequate time on planning and designing before starting to code, which I was not able to accomplish. The second one is to divide tasks and make small goals everyday such as “figure out how to show a shape on the screen” and “learn how to add random color to the shape”. The third one is to rely on lecture slides and online resources such as Stack Overflow and GeeksForGeeks to figure out where I am stuck at and what I am doing incorrectly.