Course name : Object-Oriented Programming using Java

Course code : EIE3320

student IDs/names :

19069748d Kwok Kevin

(**one-person group**)

assignment name :

Lab1 report

Class and Class Inheritance

date : 14/10/2022

Content List

Introduction

Java is the Ineternet programming and general purpose programming language. It enables users to develop and deploy applications on the Internet for servers, desktop computers , and small hand-held devices.

In this project, we are going to write a program that can compute and display the area and perimeter of circles, squares, and rectangles by understanding and applying the class and inheritance.

Objectives

- Understand the principles of Object-Oriented design.
- Apply Java in Object-Oriented software development.
- Apply UML in Object-Oriented software modelling.
- Apply Object-Oriented approach to developing computer software.
- Learn independently and be able to search for the information required in solving problems.
- Present ideas and finding effectively.

Methodology

~~How your team divides the work among the team members?~~ (One-person group)

The schedule of implementing the program

Week	Date	Scheme of Implementation
Week 6	3-8/10/2022	● Study of References ● Completion of the Classes Circle, Rectangle and Square and function ShapeTester
Week 7	10-13/10/2022	● Completion of the Class Picture and function PictureTester ● Fixing the Classes and function ● Study of References ● Programe Testing ● Completion of draw() in Circle Rectangls Square ● Interface Drawable ● Import Canvas
	14/10/2022	● Lab Report ● Programe Testing
	16/10/2022(Deadline for Submission)	● Record Demo Video ● Final Checking ● Submission of all required lab material

The program structure of the program developed

Briefing of this program

This program consists of 2 testClass and 7 classes:

- ShapeTester.java and PictureTester.java are main programe for checking whether the system can be run correctly.
- Shape is an abstract class
- Circle, Rectangle and Square are subclassese of the abstract class
- Picture is a class where the array is stored
- Drawable is an interface for implementing the visible shape

Main Function ShapeTester.java

The function creates a loop menu for user to choose different Shape(Circle/Rectenagel/Square) or shut down this program(breaking the loop). Depends of what user inputs, the main function will create a new shape object and outcome the desired results of area and parameter, and also the drawing of a desired shape in a correct size.

Function PictureTester

This function's job is to check whether various Picture class methods can operate properly and appropriately. It works by adding two each of the classes Square, Circle, and Rectangle so that users can see whether all the classes can be presented appropriately as the results.

Class Shape

The class Shape is an abstract class with two protected float variables, area and perimeter, as its member for storing the computational results generated, and four abstract methods, namely readShape(), computeArea(), computePerimeter() and displayShape(), which will be further achieve its function by the subclasses.

Class Square

The class Square is a subclass under the abstract class Shape, including a float variable length for storing the length value that provided by the end-users. There are also two constructors, Square() and Square(float l), for creating a new, special Square class if needed. After that, the four methods are included in this class, as the extension to the corresponding methods in the abstract class Shape. For the readShape() method, it is used for receiving end-users' input of the length of the square to be drawn. The computeArea() and computePerimeter() methods are respectively for the calculation of area and perimeter of the Square with the given length. The displayShape() function is served as the method for generating outputs with the results that previously computed with the other two methods. Last but not least, there is one more method draw(), which is used as drawing the corresponding circle on the canvas, with the implementation from the class Drawable.

Class Rectangle

The class Square is a subclass under the abstract class Shape, including a float variable length for storing the length and width value that provided by the end-users. There are also two constructors, Rectangel() and Rectangel(float l, float w), for creating a new, special Rectangel class if needed. After that, the four methods are included in this class, as the extension to the corresponding methods in the abstract class Shape. For the readShape() method, it is used for receiving end-users' input of the length and width of the rectangel to be drawn. The computeArea() and computePerimeter() methods are respectively for the calculation of area and perimeter of the Rectangel with the given radius. The displayShape() function is served as the method for generating outputs with the results that previously computed with the other two methods. Last but not least, there is one more method draw(), which is used as drawing the corresponding circle on the canvas, with the implementation from the class Drawable.

Class Circle

The class Circle is a subclass under the abstract class Shape, including a float variable radius for storing the radius value that provided by the end-users. There are also two constructors, Circle() and Circle(float r), for creating a new, special Circle class if needed. After that, the four methods are included in this class, as the

extension to the corresponding methods in the abstract class Shape. For the ReadShape() method, it is used for receiving end-users' input of the radius of the circle to be drawn. The computeArea() and computePerimeter() methods are respectively for the calculation of area and perimeter of the circle with the given radius. The displayShape() function is served as the method for generating outputs with the results that previously computed with the other two methods. Last but not least, there is one more method draw(), which is used as drawing the corresponding circle on the canvas, with the implementation from the class Drawable.

Class Picture

The class Picture contains an arraylist named shapes, which is used to store the shapes created,The method addShape(Shape s) is allowed to add the classes which belongs to the abstract class Shape or its three subclasses. Method computeShape() will calculate the areas and perimeters of all different classes within shapes, the arraylist. listAllShapeTypes() and listSingleShapeType are then responsible to show the stored results of classes inside shapes, but the former one will show all records and the later one will only show the results which is belonged to ClassName.

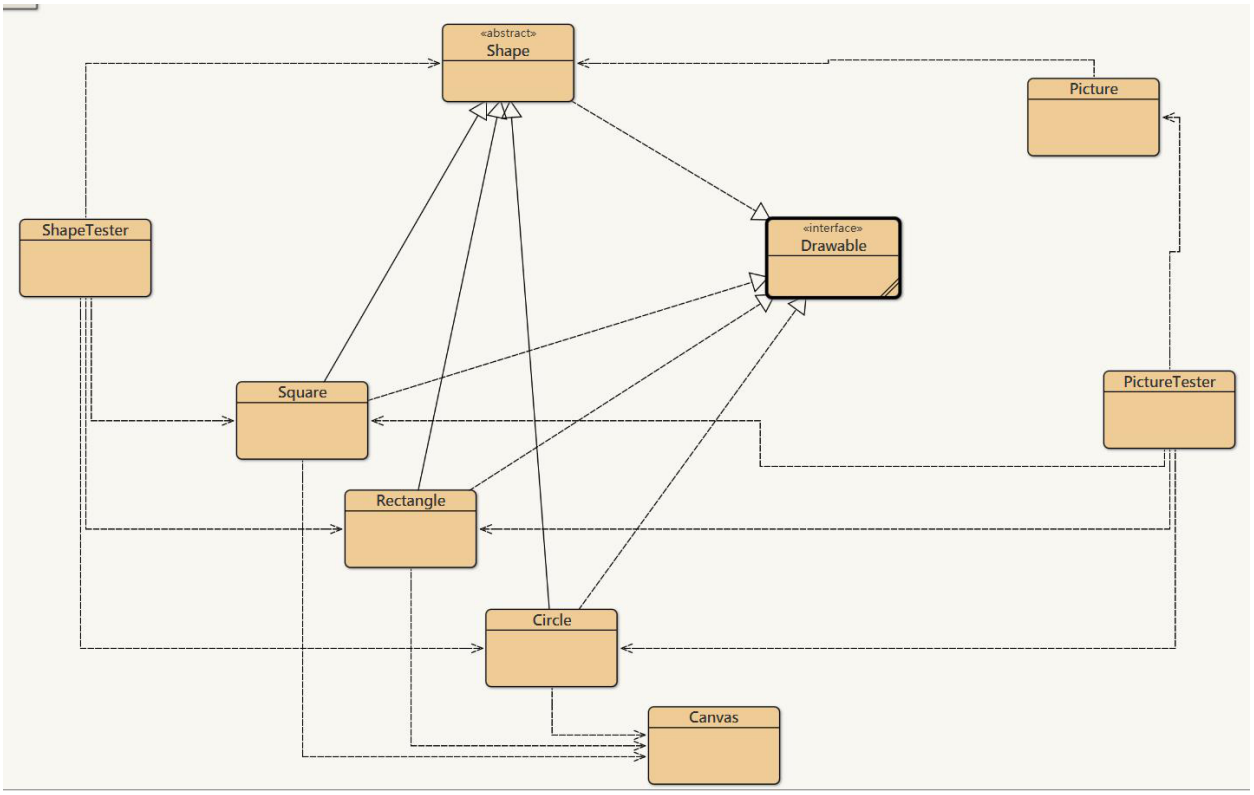
Class Canvas

The class Canvas is used for graphical display of the shapes and compat with the Shapes Demo to provide the display of results. However, as this is one of the provided class that developed by others, therefore we will not deeply discuss its content here.

Interface Drawable

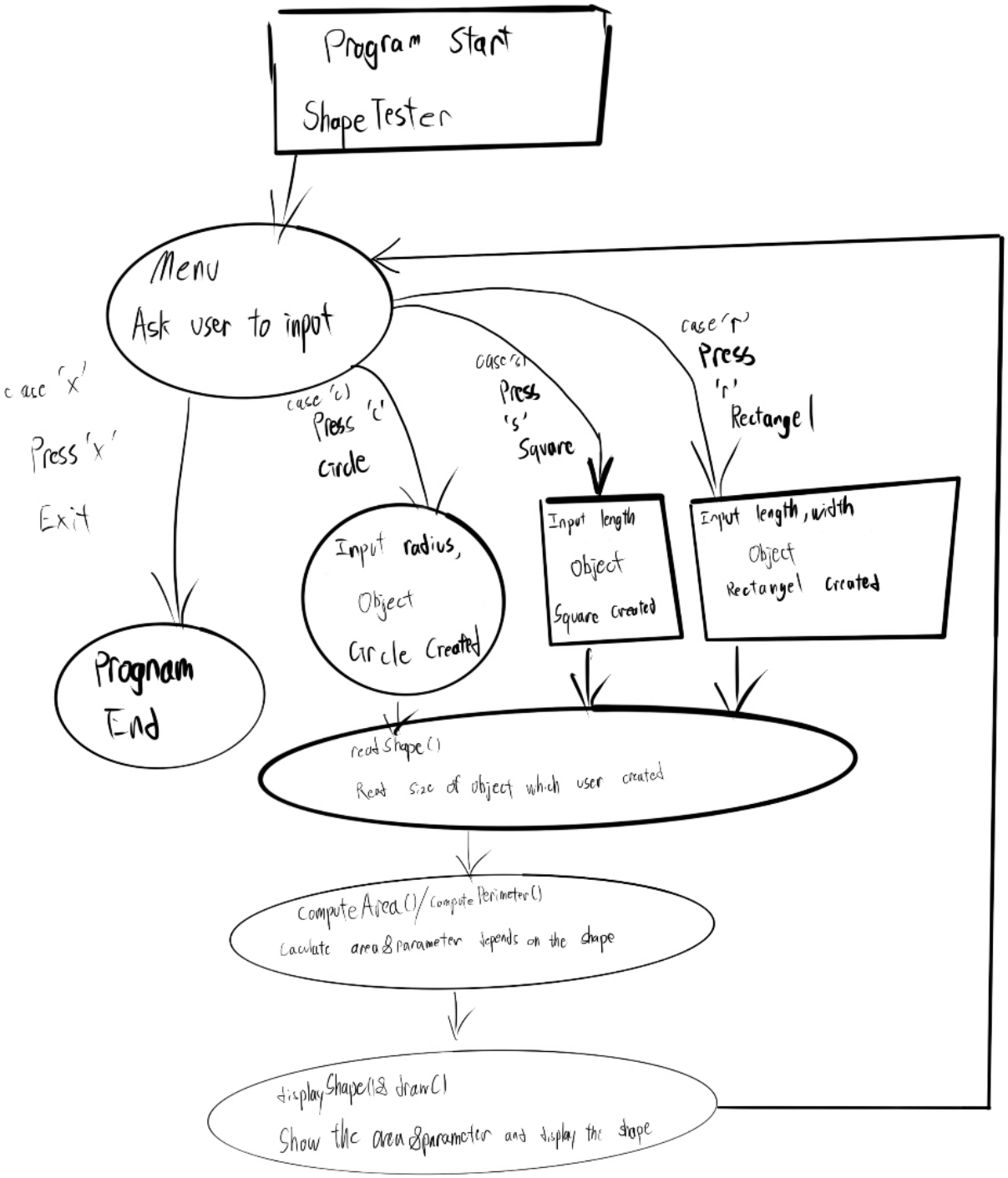
Drawable is act as an interface and with function draw(). This allow the abstract class Shape to implement so that the classes Circle, Rectangle and Square can work out the designed draw() function.

UML Diagram


«abstract»
Shape
Picture
ShapeTester
«interface»
Drawable
Square
PictureTester
Rectangle
Circle
Canvas

The flow of execution

idea flow of the program.



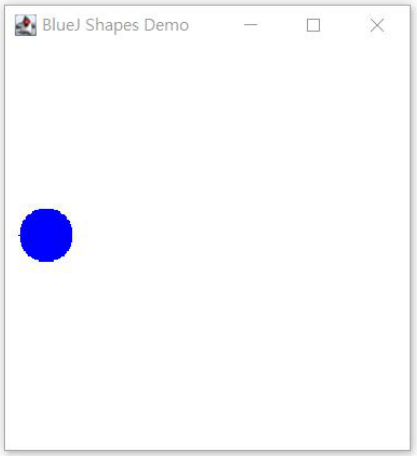
Program Test

Start program -> case ’x’ -> Program end

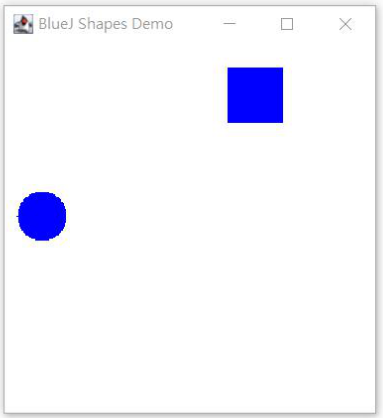
```
************************************
* Please choose one the followings:  *
* Press 'c' - Circle                 *
* Press 's' - Square                 *
* Press 'r' - Rectangle              *
* Press 'x' - EXIT                   *
************************************

x

Can only enter input while your program is running
```

Start program -> case ’c’ -> case ’s’ -> case ’r’ (Continous work)

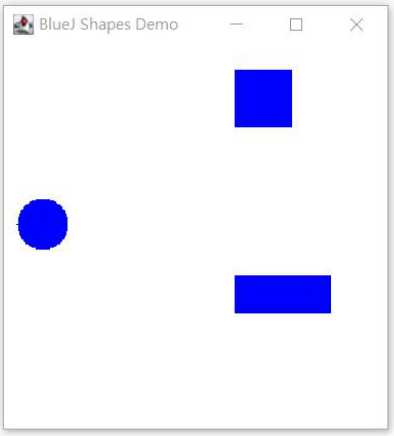
```
************************************
* Please choose one the followings:  *
* Press 'c' - Circle                 *
* Press 's' - Square                 *
* Press 'r' - Rectangle              *
* Press 'x' - EXIT                   *
************************************

c
Please input the radius:
20
Area of circle = 1256.6371
Perimeter of circle = 125.66371

************************************
* Please choose one the followings:  *
* Press 'c' - Circle                 *
* Press 's' - Square                 *
* Press 'r' - Rectangle              *
* Press 'x' - EXIT                   *
************************************
```



```
* Press 'x' - EXIT                  *
*************************************

c
Please input the radius:
20
Area of circle = 1256.6371
Perimeter of circle = 125.66371

*************************************
* Please choose one the followings:  *
* Press 'c' - Circle                 *
* Press 's' - Square                 *
* Press 'r' - Rectangle              *
* Press 'x' - EXIT                   *
*************************************

s
Please input the length:
45
Area of square = 2025.0
Perimeter of square = 180.0

*************************************
* Please choose one the followings:  *
* Press 'c' - Circle                 *
* Press 's' - Square                 *
* Press 'r' - Rectangle              *
* Press 'x' - EXIT                   *
*************************************
```




```
s
Please input the length:
45
Area of square = 2025.0
Perimeter of square = 180.0

*************************************
* Please choose one the followings:  *
* Press 'c' - Circle                 *
* Press 's' - Square                 *
* Press 'r' - Rectangle              *
* Press 'x' - EXIT                   *
*************************************

r
Please input the length:
75
Please input the width:
30
Area of rectangle = 2250.0
Perimeter of rectangle = 210.0

*************************************
* Please choose one the followings:  *
* Press 'c' - Circle                 *
* Press 's' - Square                 *
* Press 'r' - Rectangle              *
* Press 'x' - EXIT                   *
*************************************
```



Conclusion

In this laboratory project, I have recalled my code editing technique and being familiar the new language- Java in this course.

Firstly, the menu loop is an old news of the coding exercise like I learnt in pervious other coding platform(C++/Python).

Secondly, applying the usage of class like setting up the variables correctly like public, private and protect these 3 property of variables, defining public or private can also run the program successfully, correctly using the class access modifier can protect code when comes to real commercial usage. It is a good pratice for that .constructors so that I can declare the variable and class for pass objects to methods properly.

Lastly, I have acquired practice with polymorphism and inheritance, which is significant because this subject is challenging and requires more practical application before we can fully grasp it. Additionally, I strive to apply abstract classes and interfaces to decrease the complexity of the program and lighten the effort for program development.

Future Development

Limit the length of input character in the menu

If user enters more than one character in a row, the program may run error. For example, I enter a word string that starts with the letter "cc," the program may run error. This is because the user may enter different characters in the function ShapeTesting.java for the showcase of different shapes (or for exiting the program) and the scanner only char for 1 character only. Therefore, we might think about improving the checking process, limiting the input length in the menu, or changing some codings in order to check the entirety of the input String that the user has entered.

To separate diffectent Objects by fixing offset

When more than two or three forms are made, or when too many huge shapes are generated on the same canvas, some shape overlapping does happen for the present version of our application. To resolve this issue, we can add a variable to the location of the forms that will be formed while taking their sizes into account, preventing overlap. On the other hand, we may add an outline to the created forms so that even when they overlap, the object's shape can still be seen. Additionally, we may alter the color of various item shapes to enhance the overlapping effect.

References

[1]
Oracle, JavaTM Platform, Enterprise Edition 6 API Specification, 2017. [Online]. Available:
http://docs.oracle.com/javaee/6/api/ [Accessed Oct. 24, 2017].

[2]
Oracle, Interfaces, 2017. [Online]. Available: http://java.sun.com/docs/books/tutorial/java/IandI/createinterface.html [Accessed Oct. 17, 2017].

[3]
Oracle, Abstract Methods and Classes, 2017. [Online]. Available:
http://java.sun.com/
docs/books/tutorial/java/IandI/abstract.html [Accessed Oct. 17, 2017].