SCIT

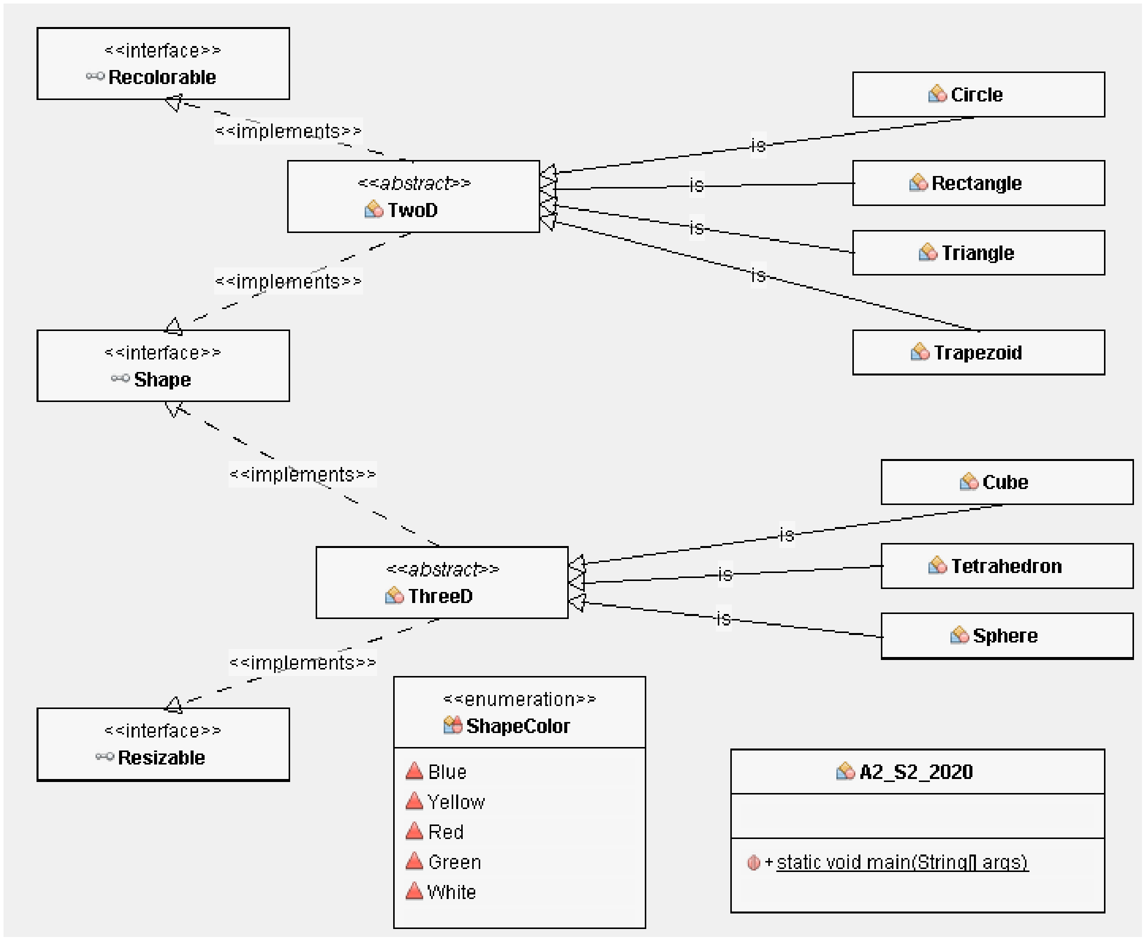
School of Computing and Information Technology
Faculty of Engineering & Information Sciences

CSIT121
Object Oriented Design and Programming
Assignment 2
File name: YourName _A2.java

Objectives:

Practice java programming with inheritance and polymorphism.

Task (5 marks)



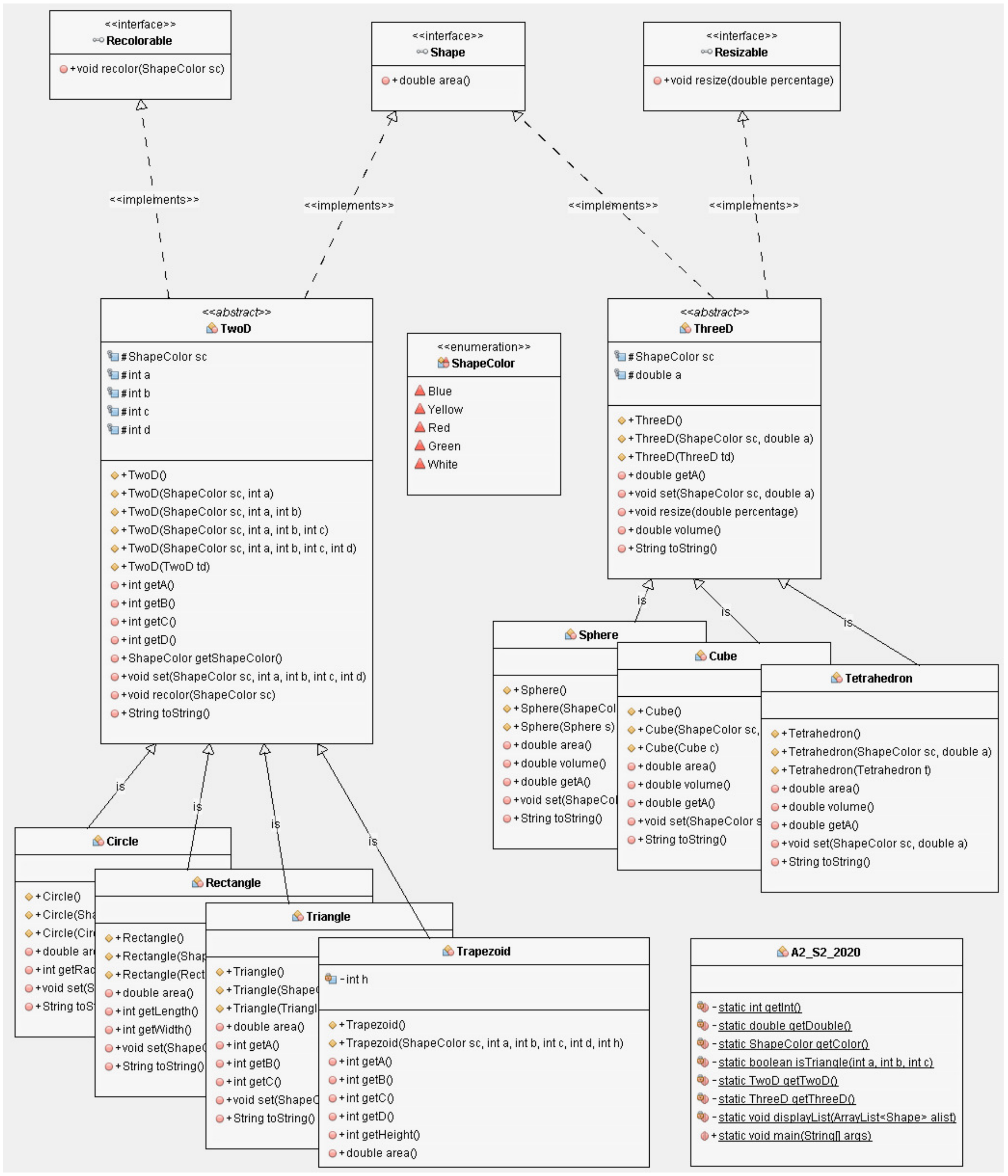
Implement the Shape hierarchy as shown in the above diagram

- Basically each `two dimensional shape` has an `area` method computes and returns the area of the two-dimensional shape.

- Each `three dimensional shape` other than the `area` (also known as surface area) method we also have a `volume` method; computes and returns, respectively, the surface area and the volume of the three-dimensional shape.

- Try to surf the internet to look for some formulas, for example, to compute the areas, the surface areas and the volumes … I did that too ☺

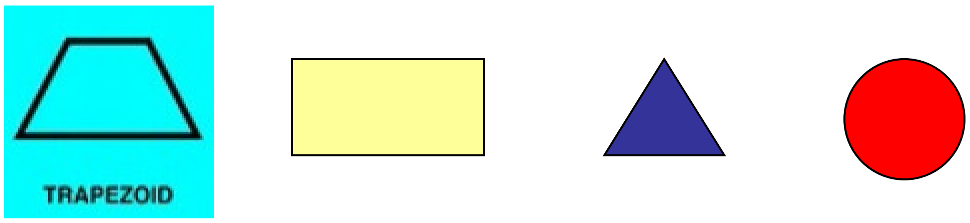
A more detailed UML diagram is shown as follow: (Note that the **#**'s before the instance variables and the methods' names mean "protected")



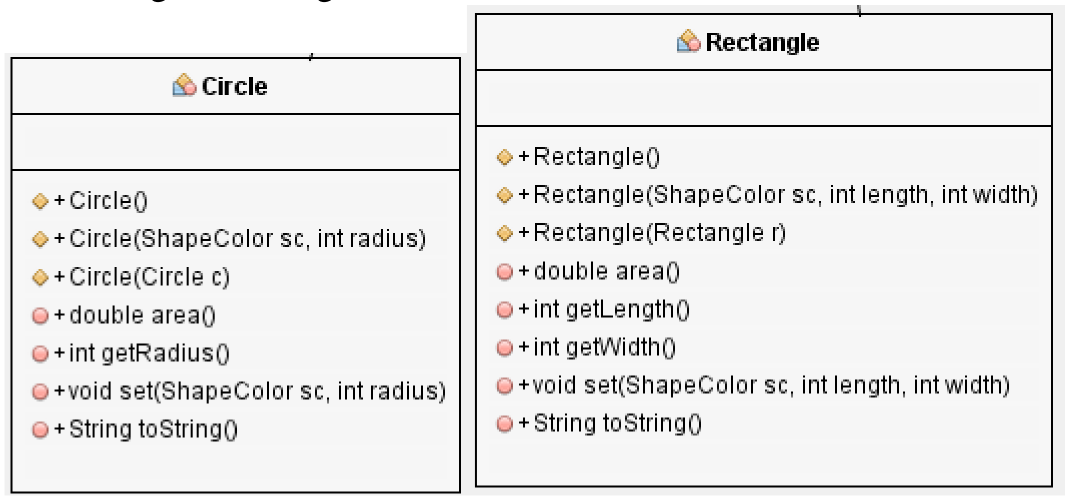
Wow … so difficult to see☺; no worry, I will break it down bit by bit and explain what you have to do …

Let us explore the above UML diagram at the highest hierarchy; you can see that `Shape` is an interface class. Two abstract classes `TwoD` and `ThreeD` implement the `Shape` interface; `TwoD` also implements `Recolorable` interface and `ThreeD` also implements `Resizable` interface and we also have an enumeration data type to describe some of the colors:

Four possible shapes for `TwoD`: one value, for example the radius, is a circle shape; two values, for example the length and the width, is a rectangle; three values for example the three sides of a triangle (provided it can be formed), five values for trapezoid (one additional value was the height, as we need that to compute the area). In this class, you can see that we have four constructors to describe the four shapes, a copy constructor, some accessor and mutator methods and a `toString` method. Each 2D shape also has a color and the color can be changed (recolor method) during runtime. You can see we also define an enumeration type to specify the color of the shapes.



The following UML diagram shows the four concrete subclasses of `TwoD`:

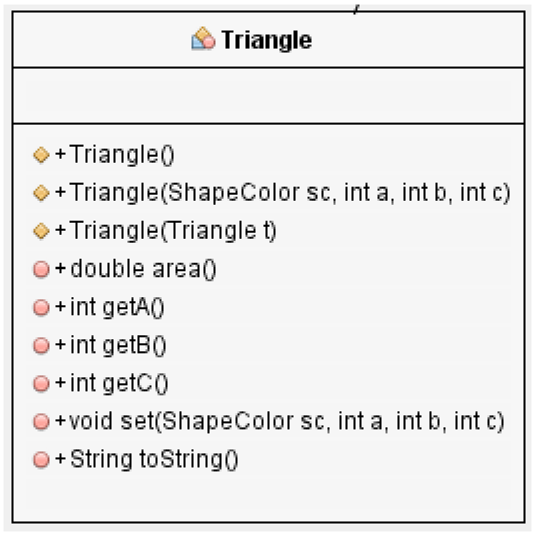


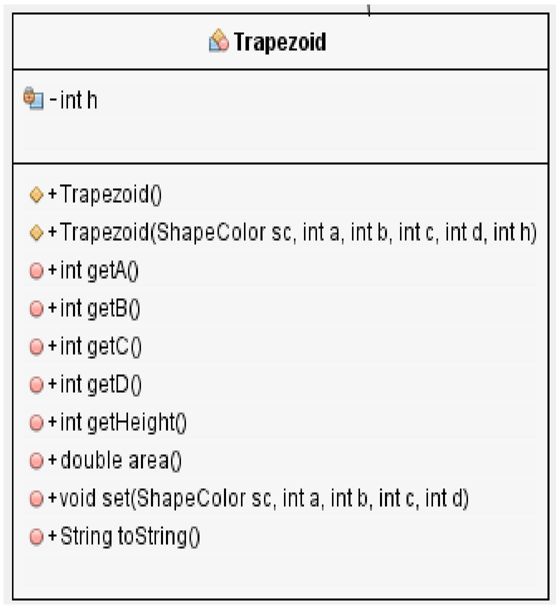
Circle

- +Circle()
- +Circle(ShapeColor sc, int radius)
- +Circle(Circle c)
- +double area()
- +int getRadius()
- +void set(ShapeColor sc, int radius)
- +String toString()

Rectangle

- +Rectangle()
- +Rectangle(ShapeColor sc, int length, int width)
- +Rectangle(Rectangle r)
- +double area()
- +int getLength()
- +int getWidth()
- +void set(ShapeColor sc, int length, int width)
- +String toString()

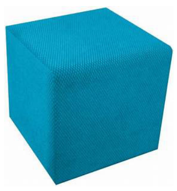




Some methods just override the super class methods (same implementations)

Information defined in each of the subclasses should be obvious in definitions.

Now, let us look at the `ThreeD` class:



Three possible shapes for `ThreeD`: Just one value can determine the shapes of a sphere, a cube and or a tetrahedron. In this class, we only also have constructors, copy constructor, accessor methods, mutator methods and a `toString` method. For a 3D object, we can compute and return the volume too. Therefore we have *one abstract method* in this abstract class `ThreeD`.

`ThreeD` class also implements `Resizable` interface class. In this Resizable interface, other than the method `resize`, to reduce the size by certain *percentage*.

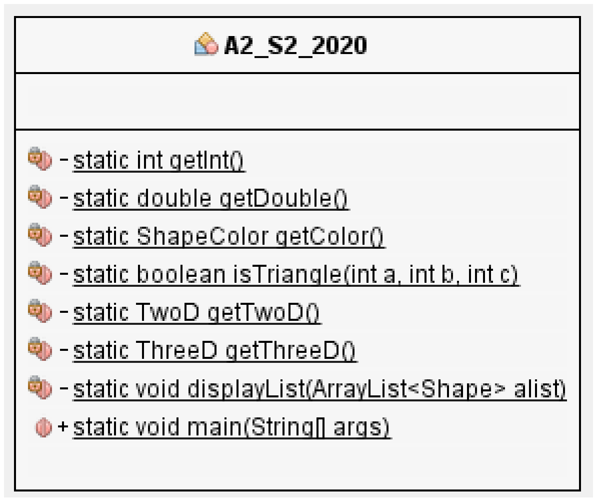
Refer to the above UML diagram for the three subclasses of `ThreeD` class: (though the diagram to small, but the methods to be designed are obvious)

Look for the surface area and volume formulas somewhere in internet to compute and to return their values, also don't forget the two private methods I have just mentioned ☺

Note that all 3D objects need to resize by a certain percentage.

Let us now explore the main class, i.e. main method is defined in this class

All shapes (2D or 3D) should be *randomly generated* and store them in an ArrayList of `Shape's`.



```
A2_S2_2020

- static int getInt()
- static double getDouble()
- static ShapeColor getColor()
- static boolean isTriangle(int a, int b, int c)
- static TwoD getTwoD()
- static ThreeD getThreeD()
- static void displayList(ArrayList<Shape> alist)
+ static void main(String[] args)
```

You can see a few private class methods are defined in this class:

- a method generates and returns a positive integer, not too large
- a method generates and returns a positive real number, not too large
- a method generates and returns a color.
- a method generates and returns a TwoD shape
- a method generates and returns a ThreeD shape
- a method to display the objects stored in the ArrayList. Note that in the display method, you display the details of each shape object, i.e. the toString method for each of the classes only display a "brief" object info, display of area / volume / resizable/ recolor should be done in this method.

Convenient to your design, minor updates to methods or additional methods are allowed.

In the main method, you *repeatedly* generate an integer `k` (0 or 1 or 2). If k is 1 you construct a 2D object; if k is 2, you construct a 3D object and k is 0, you end the task. The following shows one of the possible displays:

```
Shape 1: Cube (3D (White, 9.646))
Surface area = 558.260
Volume = 897.486
Size reduced by 24.2%: Cube (3D (White, 7.309))
Updated surface area = 320.510
Updated volume = 390.424
I am a cube shape
------------------------------
Shape 2: Trapezoid (2D (Yellow, 1, 5, 1, 7), 8)
Updated color: Red
Area = 24.000
I am a trapezoid with color changed to Red
------------------------------
Shape 3: Circle (2D (Blue, 10))
Updated color: Red
Area = 314.159
I am a Circle shape with color changed to Red
------------------------------
```

Three objects were generated and stored in an array list and you displayed the list. You can see Shape 2 and Shape 3 were 2D objects, their colors are changed during runtime (you must make sure that the color is really changed to a different color); Shape 1, a cube, its sizes, area, volume were reduced by 24.2 % (this percentage was randomly generated).

Note that the list may be empty …

IMPORTANT

Put all your classes in a file called **`YourName_A2.java`** and make sure that this file can be compiled and can be executed. Upload **ONLY** this file to Moodle**. ALL ZIP FILE SUBMISSION WILL BE REJECTED.**

No re-submission will be allowed after grading.

In the above file, remember to put down your name and also the following declaration (some similar contents):

// Tell me if it is your own work, and whether you
// have passed your program to your friends etc etc etc
// and willing to accept whatever penalty given to you.

- **Wrong file name -0.5 mark**
- **No declaration, no name etc -0.5 mark**
- **Failing to demo -1 mark**
- **Programs indentations and alignment of statements -0.5 mark**
- **Late penalty: -0.1 mark per hour.**