The City College of New York
Department of Compute Science
CSc 221: Software Design Laboratory

Assignment 4 – Fall 2019

In this assignment we write multiple Java classes in an inheritance hierarchy. As before, we will, first, rely on the Java SDK for certain tasks and design new data types while insuring maximum efficiency and reusability with fewer lines of code.

Note: please do your own work, sharing and/or copying code and/or solution ideas with/from others will result in a grade of 0 and disciplinary actions for all involved parties. If you run into any problems and have done your best to solve them, please see me before/after class or e-mail me.

Problem Description:

Implement the classes shown in the class diagram below (Figure 1). The notation used in this figure is detailed in Table 1. Please adhere to the names shown in the diagram. Some of the methods of the classes have been omitted as it is part of the assignment to decide on the best place to implement and/override certain methods. Use the provided test code and the sample output shown in Figure 2 to guide you through the class' implementations.

Interface *Comparable*:

- ✓ This is a JDK interface located under the *java.util* package. Class *Shape* implements this interface in order to define how two objects of type *Shape* should be compared when needed. We are interested in its sole *compareTo* method. Please refer to the class' documentation for detailed information.

Class *Shape*

- ✓ An abstract class which implements the *Comprable* interface and contains two abstract and non-abstract methods.
- ✓ The non-default constructor initializes the class' private fields
- ✓ *toString:* this method returns a space-delimited string of all the class' variables. i.e. the values of

  id, name, description, and *color*

Class *Shape2D* and *Shape3D*:

- ✓ Abstract classes which inherits from class Shape.
- ✓ The non-default constructor initializes the class' private fields
- ✓ *compareTo* returns *0* if two *Shapes* are equal, *-1* otherwise. Two objects of type *Shape2D* are equal if they have the same *name, height*, and *width*. For objects of *Shape3D*, the same applies as well as having the same *length*.
- ✓ *toString:* this method returns the same value as its parent class with the *height* and *width* appended. In the case of class *Shape3D*, append the value of *length*.

Class *Quadrilateral* – *Used to represent* 90° *angle quadrilateral 2D shapes*

- ✓ Inherits from class Shape2D
- ✓ *area()* is $width \times height$
- ✓ *perimeter ()* is $2 \times (width + height)$

Class *Quadrilateral3D* – *Used to represent* 90° *angle quadrilateral 3D shapes*

- ✓ Inherits from class Shape3D
- ✓ *area()* is $width \times height$
- ✓ *perimeter ()* is $2 \times (width \times height + width \times length + length \times height)$

Class *ShapeList*:

- ✓ *setShapes* points to an instance of type *TreeSet*. The set will hold objects of type Shape
- ✓ *add()*, checks if a similar *Shape* instance is already stored in the set *setShapes*. If it is, throw an exception of type Exception and message *"Duplicate object"*. If it is not, add it and return *true*. YOU MUST USE the *Contains()* method from *TreeSet* which requires that the method *compareTo* is overloaded. Do not write your own search code.

- ✓ *get2DShapes()*, returns a new set containing instances of supertype *Shape2D*. Hint: the following statement checks if the reference variable *someRefVar* points to an instance of type *MyClass*

```
if (someRefVar instanceof MyClass)
    … … …
```

- ✓ *get3DShapes()*, returns a new set containing instances of supertype *Shape3D*.
- ✓ *printFormatted()*, prints a table containing the information from the set. See Figure 2 for the output's format.

Grades:

Part of the grades for each class will be dedicated for proper logic. For example, you should reuse code from the parent class when implementing certain methods. You should also implement methods in the right place to avoid duplicate code statements.

Item	Points
Class Shape	10
Class Shape2D	10
Class Shape3D	10
Class Quadrilateral	10
Class Quadrilateral3D	10
Class ShapeList	
add	10
get2DShapes and get3DShapes	10
printFormatted	10
Correct output	10
Efficiency of code	10
	100

Class diagram legend:

Symbol	Description
	An interface
	A class
	An abstract class or method
	Private member
	Public member
	Final member
	A hollowed arrow indicates inheritance
	An open-ended arrow indicates composition
	A dotted and hollowed arrow indicates implementation

Table 1: Legend

The City College of New York
Department of Compute Science
CSc 221: Software Design Laboratory

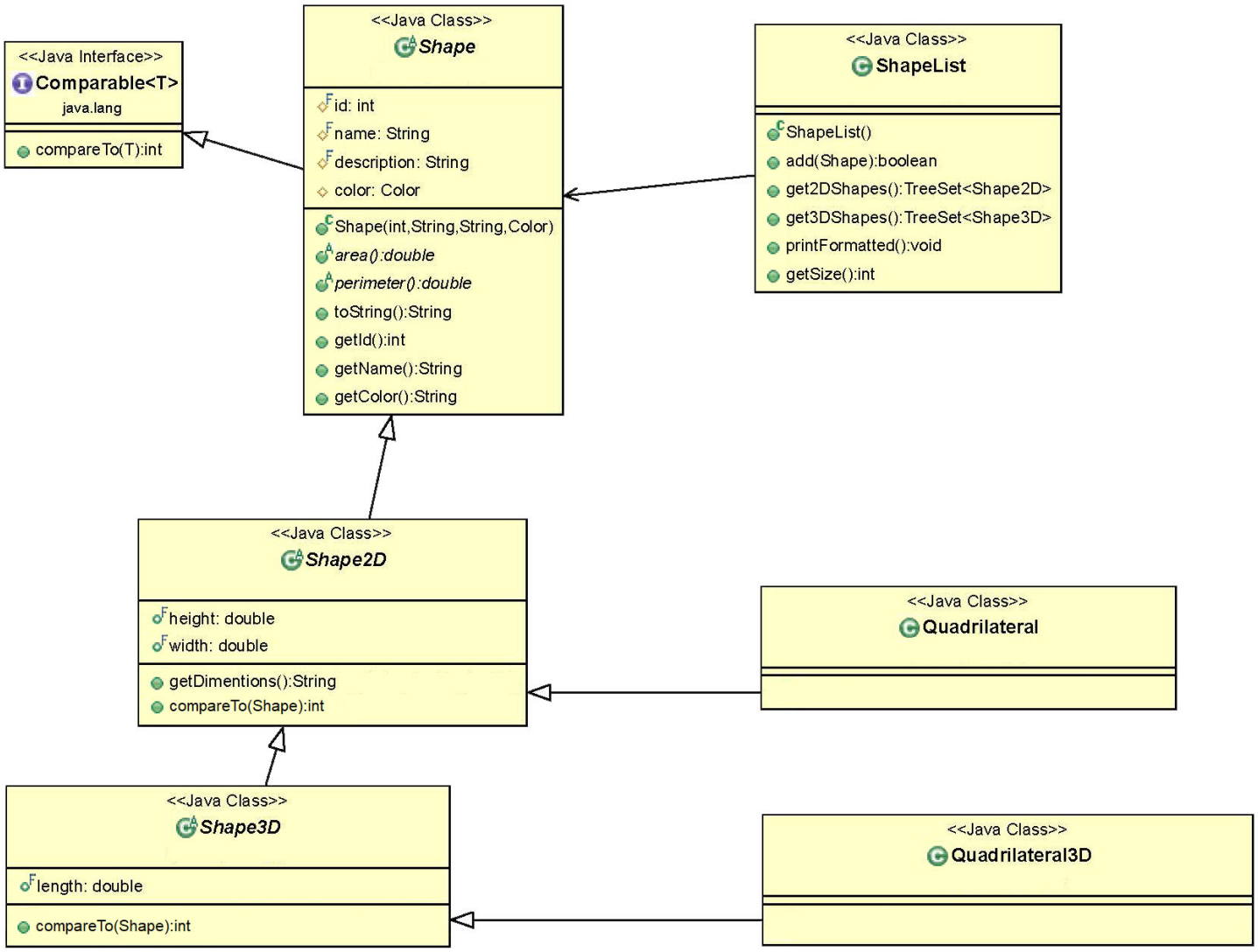

Figure 1: Class diagram

```
   Duplicate shape, skipping: 5674,Red,107.18-6.33-199.49,Cube,A red cube
   Duplicate shape, skipping: 3140,White,212.16-186.25-437.87,Cuboid,A white cuboid
Unrecognized shape, skipping: 9999,White,28.47-122.87,Triangle,An unrecognized shape
There are 15 2-Dimentional shapes
There are 13 3-Dimentional shapes
+------+-------------+-------+-----------------------+-------------------+
| ID   | Name        | Color | Dimensions            | Description       |
+------+-------------+-------+-----------------------+-------------------+
| 3140 | Cuboid      | White | $212.16:186.25:437.87 | A white cuboid    |
+------+-------------+-------+-----------------------+-------------------+
| 9149 | Cube        | Blue  | $48.03:16.1:97.68     | A blue cube       |
+------+-------------+-------+-----------------------+-------------------+
| 5678 | Cuboid      | Green | $103.96:51.11:172.19  | A green cuboid    |
+------+-------------+-------+-----------------------+-------------------+
| 8373 | Rectangle   | Green | $102.77:17.79         | A green rectangle |
+------+-------------+-------+-----------------------+-------------------+
| 9934 | Rectangle   | Blue  | $193.1:233.66         | A blue rectangle  |
+------+-------------+-------+-----------------------+-------------------+
| 5214 | Cuboid      | White | $49.59:6.81:197.2     | A white cuboid    |
+------+-------------+-------+-----------------------+-------------------+
| 8918 | Cube        | Green | $141.78:158.45:362.85 | A green cube      |
+------+-------------+-------+-----------------------+-------------------+
| 8098 | Square      | Blue  | $162.81:456.75        | A blue square     |
+------+-------------+-------+-----------------------+-------------------+
| 2210 | Square      | Blue  | $83.38:40.17          | A blue square     |
+------+-------------+-------+-----------------------+-------------------+
| 2210 | Square      | Blue  | $187.86:123.68        | A blue square     |
+------+-------------+-------+-----------------------+-------------------+
| 3076 | Square      | Red   | $467.2:395.13         | A red square      |
+------+-------------+-------+-----------------------+-------------------+
| 4583 | Square      | Green | $408.68:610.75        | A green square    |
+------+-------------+-------+-----------------------+-------------------+
| 3770 | Cuboid      | White | $118.33:283.33:33.01  | A white cuboid    |
+------+-------------+-------+-----------------------+-------------------+
| 4190 | Square      | Green | $102.43:108.75        | A green square    |
+------+-------------+-------+-----------------------+-------------------+
| 4190 | Square      | Green | $695.45:238.64        | A green square    |
+------+-------------+-------+-----------------------+-------------------+
| 9363 | Square      | Black | $62.49:159.31         | A black square    |
+------+-------------+-------+-----------------------+-------------------+
| 9478 | Cube        | Green | $423.58:178.67:169.65 | A green cube      |
+------+-------------+-------+-----------------------+-------------------+
| 5216 | Rectangle   | Green | $310.03:256.65        | A green rectangle |
+------+-------------+-------+-----------------------+-------------------+
| 5131 | Cube        | Blue  | $30.35:19.67:543.81   | A blue cube       |
+------+-------------+-------+-----------------------+-------------------+
| 5674 | Cube        | Red   | $107.18:6.33:199.49   | A red cube        |
+------+-------------+-------+-----------------------+-------------------+
| 7166 | Square      | Black | $69.29:17.09          | A black square    |
+------+-------------+-------+-----------------------+-------------------+
| 8532 | Cube        | Blue  | $12.33:410.39:128.86  | A blue cube       |
+------+-------------+-------+-----------------------+-------------------+
| 8532 | Cube        | Blue  | $48.84:121.19:427.28  | A blue cube       |
+------+-------------+-------+-----------------------+-------------------+
| 5048 | Cube        | Black | $7.16:128.98:6.92     | A black cube      |
+------+-------------+-------+-----------------------+-------------------+
| 5048 | Cube        | Black | $182.69:391.92:514.42 | A black cube      |
+------+-------------+-------+-----------------------+-------------------+
| 3051 | Square      | Red   | $296.35:249.32        | A red square      |
+------+-------------+-------+-----------------------+-------------------+
| 5352 | Square      | Black | $381.05:118.17        | A black square    |
+------+-------------+-------+-----------------------+-------------------+
| 1511 | Square      | Blue  | $52.48:86.27          | A blue square     |
+------+-------------+-------+-----------------------+-------------------+
28 rows
```

Figure 2: Test Code's Output