Student Name: Eric Vara

Total Points (10 pts) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Project: Enabling GeometricObject comparable**

Problem Description:

(*Enabling GeometricObject comparable*) Modify the GeometricObject class to implement the Comparable interface, and define a static max method in the GeometricObject class for finding the larger of two GeometricObject objects.

Draw the UML diagram and implement the new GeometricObject class.

Write a test program that uses the max method to find the larger of two circles and the larger of two rectangles.

Coding: (Copy and Paste Source Code here. Format your code using Courier 10pts)

public class Test {

// Main method

public static void main(String[] args) {

// Create two comparable circles

Circle1 circle1 = new Circle1(5);

Circle1 circle2 = new Circle1(4);

// Display the max circle

Circle1 circle = (Circle1)GeometricObject1.max(circle1, circle2);

System.out.println("The max circle's radius is " +

circle.getRadius());

System.out.println(circle);

}

}

abstract class GeometricObject1 implements Comparable {

**protected** String color; // Created a color String variable

**protected** **double** weight; // Created a weight double variable

**protected** GeometricObject1() { // Default constructor

color = "white"; // Default color

weight = 1.0; // Default weight

}

**protected** GeometricObject1(String color, **double** weight) { // Created a geometric constructor with color and weight parameters

**this**.color = color;

**this**.weight = weight;

}

**public** String getColor() { // Created getter for color

**return** color;

}

**public** **void** setColor(String color) { // Created setter for color

**this**.color = color;

}

**public** **double** getWeight() { // Created getter for weight

**return** weight;

}

**public** **void** setWeight(**double** weight) { // Created setter for weight

**this**.weight = weight;

}

**public** **abstract** **double** getArea(); // Required abstract method

**public** **abstract** **double** getPerimeter(); // Required abstract method

**public** **int** compareTo(GeometricObject1 a) { // Created compareTo method

**if** (getArea() < a.getArea())

**return** -1;

**else** **if** (getArea() == a.getArea())

**return** 0;

**else**

**return** 1;

}

**public** **static** GeometricObject1 max(GeometricObject1 a1, GeometricObject1 a2) { // Created a max method with 2 compare parameters

**if** (a1.compareTo(a2) > 0)

**return** a1;

**else**

**return** a2;

}

}

// Circle.java: The circle class that extends GeometricObject

class Circle1 extends GeometricObject1 {

**protected** **double** radius;

**public** Circle1() { // Created default constructor

**this**(1.0, "white", 1.0);

}

**public** Circle1(**double** radius) { // Created a circle constructor with

color parameters

**this**.radius = radius;

}

**public** Circle1(**double** radius, String color, **double** weight) { // Created

a circle constructor with radius, color, and weight parameters

**this**.radius = radius;

**this**.color = color;

**this**.weight = weight;

}

**public** **double** getRadius() { // Created getter for radius

**return** radius;

}

**public** **void** setRadius(**double** radius) { // Created setter for radius

**this**.radius = radius;

}

**public** **double** getArea() { // Created getter for area

**return** radius \* radius \* Math.***PI***;

}

**public** **double** getPerimeter() { // Created getter for perimeter

**return** 2 \* radius \* Math.***PI***;

}

**public** **boolean** equals(Circle1 circle) { // Override the equals() method

defined in the Object class

**return** **this**.radius == circle.getRadius();

}

@Override

**public** String toString() {

**return** "Circle radius = " + radius;

}

@Override

**public** **int** compareTo(GeometricObject1 a) {

**if** (getRadius() > ((Circle1) a).getRadius())

**return** 1;

**else** **if** (getRadius() < ((Circle1) a).getRadius())

**return** -1;

**else**

**return** 0;

}

}

Submit the following items:

1. Print this Word file and Submit to the DropBox for this assignment.

2. Compile, Run, and Submit to Schoology (you must submit the program regardless whether it is complete or incomplete, correct or incorrect)