CSE 213 – Homework 2 Rubric

Student Name:marisa loraas

Grader Name: Mason Bentley

Problem	Score	Total
Style Guidelines (7 points)		**7**
Submission is named cse213_<firstname>_<lastname>_hw2.tar.gz	1	1
Packages are named: oop.<firstinitial><lastname>.hw1.<number> where <number> is the problem number	1	1
Code follows Google's style guide reasonably well, and uses four-space indentation	5	5
Point.java		**15**
Three constructors: 1. **new Point()** sets all attributes to 0 2. **new Point(x,y)** sets the Euclidean coordinates 3. **new Point(point)** creates a copy of the given point	2	3
SetX() and setY() correctly assign the polar coordinates to **radius** and **angle**	3	3
setRadius() and **setAngle()** correctly assign the Euclidean coordinates to x and y	3	3
compareTo() returns 1. 0 when x = other.x and y = other.y 2. -1 when x ≤ other.x and y ≤ other.y 3. 1 when x > other.x or y y > other.y	4	4
toString() formats the point as "(x,y)"	2	2
Rectangle.java		**23**
n**ew Rectangle()** creates a rectangle with the lower left at (0, 0) and the upper right at (1, 1)	3	3
New Rectangle()(lowPoint, highPoint) checks that 1. **lowPoint.x ≤ highPoint.x** 2. **lowPoing.y ≤ highPoint.y** If both are true, it assigns both attributes with new copies of the given points. If both are false, it does the same with swapped input arguments (i.e. **lowerLeft = new Point(highPoint)** and vice versa).	6	8

If one is true but the other is false, it constructs points for the opposite corners of the rectangle and sets the attributes accordingly.		
new Rectangle(width, height) creates a rectangle with one corner on (0,0) and the opposite corner on (width, height)	3	3
getLowerLeft() and getUpperRight() return new copies of the respective points	2	3
area() returns w x y	2	2
perimeter() returns 2w + 2h	2	2
inBounds() checks that **lowerLeftX ≤ x ≤ upperRightX**, and **LowerLeftY ≤ y ≤ upperRightY**	2	2
Circle.java		**19**

Two constructors:
1. **new Circle()** sets the center to (0, 0) and the radius to 1
2. **new Circle(point, radius)**
  1. sets the center to a new **copy** of the given point
  2. sets the radius to the **absolute value** of the given value
	3	3

setCenter() creates a new copy of the given point	3	3
setRadius() sets the absolute value of the argument	3	3
area() returns pi * r^2	2	2
perimeter() returns 2*pi*r	2	2
inBounds() checks that the distance from the given point to the center is ≤ the radius	2	2
getBoundingBox() creates a rectangle where: 1. *lowerLeft.x = center.x - radius* 2. *lowerLeft.y = center.y - radius* 3. *upperRight.x = center.x + radius* 4. *upperRight.y = center.y + radius*	4	4
Temperature.java Note: A mistake in the assignment says to convert Celsius to Kelvin with the formula K = C - 273.15; the correct formula is K = C + 273.15. Either version should be considered correct, since the assignment was misleading		**16**
The Degrees enumeration (either in its own file or public within the Temperature class) differentiates Kelvin, Celsius, and Fahrenheit	1	1
One constructor assumes its input is in degrees Celsius, sets the **celsius** attribute to the given value, and sets the other two attributes to the correctly converted values	2	2

The second constructor takes an instance of th Degrees enumeration, uses it to decide which of the three attributes to set with the given value, and sets all three attributes correctly	3	3
setKelvin() converts its input to Celsius and Fahrenheit, and sets all three attributes correctly	1	3
setCelsius() converts its input to Kelvin and Fahrenheit, and sets all three attributes correctly	1	3
setFahrenheitt() converts its input to Kelvin and Celsius, and sets all three attributes correctly	1	3
toString() shows thetemperature in degrees Celsius, to two decimal places (e.g. "13.37 C")	1	1
Converter.java		8
Prompts user for a double, and two units (K, C, or F)	0	2
Prints an error message when given invalid input, and prompts the user for a correction	2	3
Prints the correctly converted temperature	0	3
WindChill.java		12
The program imports **oop.student.hw2.two.Temperature**	1	1
windChill() gets the temperature in degrees Fahrenheit, and returns a temperature whose Fahrenheit value is: 35.74 + 0.6215 * t - 35.3225 * v^0.16	0	3
frostBite() gets the temperature in degrees Celsius, and returns: (-26.133 * v + 1994.6) * (-4.8 - t)^-1.668	0	3
frostBite() returns -1 if the temperature is ≥ -15°C; and returns 0 if the temperature is ≤ 50°C or the wind speed is greater than 50	0	3
main() prompts the user for a temperature, unit, and wind speed, passes them to the other methods, and prints the results	2	2
Total Score	78	**100**
Comments: Wind chill program produces wrong output, as does the converter. (both wrong because of temperature.java)		