***App.java***

package app;

/\*\*

\*

\* @custom.application\_name Lab\_2\_GeometricObject

\* @custom.class\_name App

\*

\* @custom.author Daniel C. Landon Jr.

\* @custom.instructor Dr. Bob Walsh

\* @custom.course CSCI 202 - Introduction to Software Systems

\* @custom.date\_started 02.04.2020

\* @custom.date\_due 02.20.2020

\*

\* @custom.class\_notes None

\*

\* @custom.pre\_condition None

\*

\* @custom.post\_condition None

\*

\* @custom.javadoc\_tags In order to use @custom.tag\_name in javadocs you must include the folloinwing in the command line to generate the docs. This part must be after you have indicated what files to process;

\*

\* ' -tag custom.tag\_name:a:"tag\_name" '

\*

\* The first part identifies the tag in the code, the second part in quotes indentifies what will be printed in the javadocs when they are generated. If you do not include this in the command to generate the docs you will get an error/warning.

\*

\*/

public class App {

/\*\*

\*

\* @custom.method\_name main

\*

\* @custom.author Daniel C. Landon Jr.

\* @custom.date\_started 02.04.2020

\*

\* @custom.method\_notes none

\*

\* @custom.pre\_condition Interfaces and Abstract class must exist

\*

\* @custom.post\_condition none

\*

\* @param args command line arguments

\* @throws Exception go figure

\*/

public static void main(String[] args) throws Exception {

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Circle \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

Circle \_circleOne = new Circle();

Circle \_circleTwo = new Circle(2.1);

System.out.println("Area of Circle \_circleTwo is "

+ \_circleTwo.getArea());

System.out.println("Perimeter of Circle \_circleTwo is "

+ \_circleTwo.getPerimeter());

if (\_circleOne.compareTo(\_circleTwo) == 0) {

System.out.println("Circle \_circleOne and \_circleTwo have equal coverage of area"); } // end if

else if (\_circleOne.compareTo(\_circleTwo) > 0) {

System.out.println("Circle \_circleOne has larger area than the circle \_circleTwo"); } // end else if

else {

System.out.println("Circle \_circleOne has smaller area than the circle \_circleTwo"); } // end else

Circle \_circleThree = \_circleTwo.clone();

if (\_circleTwo.compareTo(\_circleThree) == 0) {

System.out.println("Circle \_circleTwo and \_circleThree have equal coverage of area"); } // end if

else if (\_circleTwo.compareTo(\_circleThree) > 0) {

System.out.println("Circle \_circleTwo has larger area than the circle \_circleThree"); } // end else if

else {

System.out.println("Circle \_circleTwo has smaller area than the circle \_circleThree"); } // end else

System.out.println("Circle \_circleThree -->" + \_circleThree);

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Ellipse \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

Ellipse ellipse1 = new Ellipse(2.0,2.0);

Ellipse ellipse2 = new Ellipse(3.0, 3.0);

System.out.println("Area of Ellipse ellipse2 is "

+ ellipse2.getArea());

System.out.println("Perimeter of Ellipse ellipse2 is "

+ ellipse2.getPerimeter());

if (ellipse1.compareTo(ellipse2) == 0) {

System.out.println("Ellipse ellipse1 and ellipse2 have equal coverage of area"); } // end if

else if (ellipse1.compareTo(ellipse2) > 0) {

System.out.println("Ellipse ellipse1 has larger area than the circle ellipse2"); } // end else if

else {

System.out.println("Ellipse ellipse1 has smaller area than the ellipse ellipse2"); } // end else

Ellipse ellipse3 = ellipse2.clone();

if (ellipse2.compareTo(ellipse3) == 0) {

System.out.println("Ellipse ellipse2 and ellipse3 have equal coverage of area"); } // end if

else if (ellipse2.compareTo(ellipse3) > 0) {

System.out.println("Ellipse ellipse2 has larger area than the ellipse ellipse3"); } // end else if

else {

System.out.println("Ellipse ellipse2 has smaller area than the ellipse ellipse3"); } // end else

System.out.println("Ellipse ellipse3: " + ellipse3);

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Octagon \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

Octagon octagon1 = new Octagon();

Octagon octagon2 = new Octagon(3.0);

System.out.println("Area of Octagon octagon2 is "

+ octagon2.getArea());

System.out.println("Perimeter of Octagon octagon2 is "

+ octagon2.getPerimeter());

if (octagon1.compareTo(octagon2) == 0) {

System.out.println("Octagon ooctagon1 and octagon2 have equal coverage of area"); } // end if

else if (octagon1.compareTo(octagon2) > 0) {

System.out.println("Octagon octagon1 has larger area than the Octagon octagon2"); } // end else if

else {

System.out.println("Octagon octagon1 has smaller area than the Octagon octagon2"); } // end else

Octagon octagon3 = octagon2.clone();

if (octagon2.compareTo(octagon3) == 0) {

System.out.println("Octagon octagon2 and octagon3 have equal coverage of area"); } // end if

else if (octagon2.compareTo(octagon3) > 0) {

System.out.println("Octagon octagon2 has larger area than the Octagon octagon3"); } // end else if

else {

System.out.println("Octagon octagon2 has smaller area than the Octagon octagon3"); } // end else

System.out.println("Octagon octagon3: " + octagon3);

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Equaleteral Triangle \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

EquilateralTriangle et1 = new EquilateralTriangle();

EquilateralTriangle et2 = new EquilateralTriangle(3.0);

System.out.println("Area of Equilateral Triangle et2 is "

+ et2.getArea());

System.out.println("Perimeter of Equilateral Triangle et2 is "

+ et2.getPerimeter());

if (et1.compareTo(et2) == 0) {

System.out.println("Equilateral Triangle et1 and et2 have equal coverage of area"); } // end if

else if (et1.compareTo(et2) > 0) {

System.out.println("Equilateral Triangle et1 has larger area than the Equilateral Triangle et2"); } // mend else if

else {

System.out.println("Equilateral Triangle et1 has smaller area than the Equilateral Triangle et2"); } // end else

EquilateralTriangle et3 = et2.clone();

if (et2.compareTo(et3) == 0) {

System.out.println("Equilateral Triangle et2 and et3 have equal coverage of area"); } // end if

else if (et2.compareTo(et3) > 0) {

System.out.println("Equilateral Triangle et2 has larger area than the Equilateral Triangle et3"); } // end else if

else {

System.out.println("Equilateral Triangle et2 has smaller area than the Equilateral Triangle et3"); } // end else

System.out.println("Equilateral Triangle et3: " + et3);

} // end main

} // end App

***Comparable.java***

package app;

/\*\*

\* Comparable

\*/

public interface Comparable {

public int compareTo(Object obj);

}

***Eccentric.java***

package app;

public interface Eccentric{

double eccentricity();

}

***GeometricObject.java***

package app;

/\*\*

\* The

\* <code>GeometricObject</code> class is the super class of all geometric shapes

\* in this package. Derived classes must implement getArea and getPerimeter.

\*

\* @author Daniel Liang

\* @since Spring 2013

\*/

public abstract class GeometricObject {

private String color = "white";

private boolean filled;

private java.util.Date dateCreated;

/\*\*

\* Construct a default geometric object for implicit invocation. Sets

\* creation date of this geometric object

\*

\*/

protected GeometricObject() {

dateCreated = new java.util.Date();

}

/\*\*

\* Construct a geometric object with color and filled value Sets creation

\* date of this geometric object

\*

\* @param color : color of this geometric object

\* @param filled : is this object is filled or not.

\*/

protected GeometricObject(String color, boolean filled) {

dateCreated = new java.util.Date();

this.color = color;

this.filled = filled;

}

/\*\*

\* @return a string representation of this object

\*/

public String toString() {

return "created on " + dateCreated + "\ncolor: " + color

+ " and filled: " + filled;

}

/\*\*

\* Abstract method getArea. Must be implemented by sub classes of

\* GeometricObject

\*

\* @return area of this geometric object

\*/

public abstract double getArea();

/\*\*

\* Abstract method getPerimeter. Must be implemented by sub classes of

\* GeometricObject

\*

\* @return perimeter of this geometric object

\*/

public abstract double getPerimeter();

}

***Circle.java***

package app;

public class Circle extends Ellipse {

private double radius = 0.0;

public Circle() {

super(1.0, 1.0);

radius = 1.0;

} // end Circle constructor

public Circle(double radius) {

super(radius,radius);

System.out.println("This circle has a radius of: " + radius);

this.radius = radius;

} // end Circle constructor

public double getRadius() { return radius; } // end getRadius

public void setRadius(double radius) { this.radius = radius; } // end setRadius

public double getArea() { return radius \* radius \* Math.PI; } // end getArea

public double getPerimeter() { return 2 \* radius \* Math.PI; } // end getPerimeter

@Override

public String toString() { return "[Circle] radius = " + radius; } // end toString

@Override

public int compareTo(Object obj) {

if (this.getArea() > ((Circle) obj).getArea()) { return 1; }

else if (this.getArea() < ((Circle) obj).getArea()) { return -1; }

else { return 0; }

} // end compareTo

@Override

public boolean equals(Object obj) { return this.radius == ((Circle) obj).radius; } // end equals

@Override

public Circle clone() {

System.out.println("Getting Circle to clone...");

return (Circle)super.clone();

} // end clone

}

***Ellipse.java***

package app;

// Complete all methods

// Add JavaDoc with explanations.

// most code here supplied by instructor

public class Ellipse extends GeometricObject implements Eccentric, Comparable, Cloneable {

double a = 0.0;

double b = 0.0;

public Ellipse(double s1, double s2) {

if(s1 < s2) {

a = s2;

b = s1;

}

else {

a = s1;

b = s2;

}

} // end Ellipse constructor

@Override

public double getPerimeter() {

return (Math.PI) \* (Math.sqrt(2 \* (Math.pow(a,2) + Math.pow(b,2) + (a - b) / 2)));

} // end getPerimeter

@Override

public double getArea() {

return(Math.PI \* a \* b);

} // end getArea

public double perimeter()

{

//method body missing

System.out.println("perimeter");

return 0;

} // end perimeter

public double area()

{

//method body missing

System.out.println("area");

return 0;

} // end area

public double eccentricity() {

double e = 0.0;

e = Math.sqrt(a \* a + b \* b) / a;

return e;

} // end eccentricity

public String toString() {

return "Ellipse Perimeter: " + getPerimeter() + "\nArea: " + getArea() + "\n";

} // end toString

@Override

public int compareTo(Object obj) {

if (this.getArea() > ((Ellipse) obj).getArea()) { return 1; } // end if

else if (this.getArea() < ((Ellipse) obj).getArea()) { return -1; } // end else if

else { return 0; } // end else

} // end compareTo

@Override

public Ellipse clone(){

try{

System.out.print("Getting Ellipse to clone...");

return(Ellipse) super.clone();

}//try

catch(Exception e){

System.out.println("UH-OH in Ellipse");

return null;

}//catch

}//clone

} // end Ellipse

***EquilaterialTriangle.java***

package app;

/\* Assumes a proper triangle.

\* EquilateralTriangle is a GeometricObject.

\* EquilateralTriangles are Comparable and Cloneable

\* Must contain Overloaded constructors

\* Add Javadoc as shown in Circle class

\* Remove all comment lines added by Dr.H.

\*/

public class EquilateralTriangle extends GeometricObject implements Comparable, Cloneable {

double side = 0.0;

public EquilateralTriangle() { this.side = 1.0; } // end EquilateralTriangle

public EquilateralTriangle(double a) { side = a; } // end EquilateralTriangle

@Override

public double getPerimeter() { return (side \* 3); } // end getPerimeter

@Override

public double getArea() { return ((side \* side \* Math.sqrt(3)) / 4 ); } // end getArea

@Override

public String toString(){ return "Equilateral Triangle Perimeter: " + getPerimeter() + "\nArea: " + getArea() + "\n"; } // end toString

@Override

public int compareTo(Object obj) {

if (this.getArea() > ((EquilateralTriangle) obj).getArea()) { return 1; } // end if

else if (this.getArea() < ((EquilateralTriangle) obj).getArea()) { return -1; } // end else if

else { return 0; } // end else

} // end compareTo

@Override

public EquilateralTriangle clone(){

try{

System.out.print("Getting EquilateralTriangle to clone...");

return(EquilateralTriangle) super.clone();

} // end try

catch(Exception e){

System.out.println("UH-OH in EquilateralTriangle");

return null;

} // end catch

} // end clone

} // end EquilateralTriangle

***Octagon.java***

package app;

public class Octagon extends GeometricObject implements Comparable, Cloneable {

private double side = 0.0;

public Octagon(){ this.side = 1.0; } // end Octagon constructor

public Octagon(double side){ this.side = side; } // end Octagon constructor

@Override

public double getArea() { return (2 + 4 / Math.sqrt(2)) \* side \* side; } // end getArea

@Override

public double getPerimeter() { return 8 \* side; } // end getPerimeter

@Override

public int compareTo(Object obj) {

if (this.getArea() > ((Octagon) obj).getArea()) { return 1; } // end if

else if (this.getArea() < ((Octagon) obj).getArea()) { return -1; } // end else if

else { return 0;} // end else

} // end compareTo

public Octagon clone(){

try{

System.out.print("Getting Ellipse to clone...");

return(Octagon) super.clone();

} // end try

catch(Exception e){

System.out.println("UH-OH in Octagon");

return null;

} // end catch

} // end clone

} // end Octagon

***CONSOLE OUTPUT***

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Circle \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

This circle has a radius of: 2.1

Area of Circle \_circleTwo is 13.854423602330987

Perimeter of Circle \_circleTwo is 13.194689145077131

Circle \_circleOne has smaller area than the circle \_circleTwo

Getting Circle to clone...

Getting Ellipse to clone...Circle \_circleTwo and \_circleThree have equal coverage of area

Circle \_circleThree -->[Circle] radius = 2.1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Ellipse \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Area of Ellipse ellipse2 is 28.274333882308138

Perimeter of Ellipse ellipse2 is 18.84955592153876

Ellipse ellipse1 has smaller area than the ellipse ellipse2

Getting Ellipse to clone...Ellipse ellipse2 and ellipse3 have equal coverage of area

Ellipse ellipse3: Ellipse Perimeter: 18.84955592153876

Area: 28.274333882308138

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Octagon \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Area of Octagon octagon2 is 43.45584412271571

Perimeter of Octagon octagon2 is 24.0

Octagon octagon1 has smaller area than the Octagon octagon2

Getting Ellipse to clone...Octagon octagon2 and octagon3 have equal coverage of area

Octagon octagon3: created on Wed Feb 26 01:52:18 EST 2020

color: white and filled: false

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Equaleteral Triangle \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Area of Equilateral Triangle et2 is 3.8971143170299736

Perimeter of Equilateral Triangle et2 is 9.0

Equilateral Triangle et1 has smaller area than the Equilateral Triangle et2

Getting EquilateralTriangle to clone...Equilateral Triangle et2 and et3 have equal coverage of area

Equilateral Triangle et3: Equilateral Triangle Perimeter: 9.0

Area: 3.8971143170299736