**OBJECT ORIENTED PROGRAMMING**

**LAB# 12 TASKS**

## **Implementation Of Interfaces**

Below we consider an extension to our Shape class. Here we discuss about elliptical shapes which have properties in addition to area and perimeter which are of interest to us.

Consider the Shape called Ellipse. An ellipse is basically nothing but an elongated circle. A circle has a radius. An ellipse has two measurements: the major axis (*a*) and the minor axis (*b*)

with *a* > *b*.

radius

Major axis (*a*)

Minor axis (*b*)

Note that in an ellipse if *a* = *b* then it becomes a circle.

The measure of “roundness” of an ellipse is given by its eccentricity *e* where the value of *e* is always between 0 and 1. For a circle *e* = 0. The higher the value of *e*, the higher is the deviation of the ellipse from its roundness.

Eccentricity is associated with a lot of other shapes as well e.g. hyperbola, parabola, etc.

The various formulae for the ellipse are:

Perimeter = P = π [Note that if *a* = *b* = *r*, then P = 2π*r*]

Area = A = π*ab*

Eccentricity = *e* =

We can define an interface called Eccentric for all eccentric Shapes.

**Eccentric.java**

/\*

\* Eccentric.java

\*/

interface Eccentric {

double eccentricity();

}

Note here that no access modifiers have been given for the method eccentricity. The reason is that all methods in an interface are by default public and abstract. Now we can define our class Ellipse.

#### Ellipse.java

/\*

\* Ellipse.java

\*/

public class Ellipse extends Shape implements Eccentric

{

double a, b;

public Ellipse(double s1, double s2){

if(s1 < s2) {

a = s2;

b = s1;

}

else {

a = s1;

b = s2;

}

}

public double perimeter(){

//method body missing

}

public double area(){

//method body missing

}

public double eccentricity(){

//method body missing

}

public String toString(){

//method body missing

}

}

1. By looking at the formulae for an ellipse, provide the missing code for all of the methods in the class Ellipse including the toString() method. Test your program by calling the methods of all eccentric shapes. Your output should look as follows (for an ellipse with *a* = 10 and *b* = 7)
2. Square
3. Area=100.0
4. Perimeter=40.0
5. Ellipse
6. Area=219.9114857512855
7. Perimeter=53.8212680240788
8. Eccentricity=0.714142842854285
9. Press any key to continue...

How about the following class Circle. Since a Circle is a special case of an Ellipse, will the output of TestShapes.java be affected if the following class is used instead of the class Circle used previously:

##### *Circle.java*

public class Circle extends Ellipse {

public Circle(double radius){

super(radius, radius);

}

}

With this modification, the class diagram would look as follows:

