**The Shape Classes Assignment**

Put the definition of the three classes below in a file named *shapes.py* and submit that file.

**1**.

You are to design a Point class that models a point in the plane using Cartesian coordinates, which will be given by data attributes x and y. Your class should include a method named translate with data parameters dx and dy that are used to move the point dx units along the x-axis and dy units along the y-axis. You should also have a method named dist that computes the distance between the Point and another Point given as a parameter to the method. You should also overload the special method that returns a string representation of a Point object so that the following code works as shown.

>>> P = Point(2.5,3.6)  
>>> print("Here is my point: %s" %P)  
Here is my point: (2.5,3.6)

**2.**

We next consider ellipses whose axes are parallel to the x and y axes of the coordinate plane. An ellipse is determined by the distance A from the center to the highest point on the ellipse boundary along the vertical line through the center; and the distance B from the center to the point on the ellipse boundary to the right of the center along the horizontal line through the center. The area of the ellipse is then given by πA2 + πB2. The height and width of the ellipse are 2\*A and 2\*B, respectively.

You are to create an Ellipse class with a Point data member named center; and float data members x\_extent and y\_extent that correspond the constants A and B described above.

Your class should have get\_area and translate methods; translate has two data attributes dx and dy and simply moves the center of the ellipse . You should also overload the special method that returns a string representation of an Ellipse object so that the following code works as shown.

>>> P = Point(1.0,3.0)  
>>> E = Ellipse(P,2.3,1.2)  
>>> print E  
Ellipse with Center: (1.0,3.0); Width: 2.4; Height: 4.6

You should also define a function getEllipse() that prompts the user for the appropriate information for an Ellipse, then returns that Ellipse object. This function is not a class method.

**3.**  
Now define a subclass Circle of Ellipse to represent a circle. We should be able to print a circle, get the area of the circle, move a circle and decide if one circle is inside another. Use inheritance wherever it is possible and natural to do so. Here is the first part of the contains method.

**def contains(self,other):  
 ''' returns True if and only if circle other is completely  
 contained inside circle self  
 '''  
 *# complete the code; use pictures to guide you work;  
 # it may help to consider the line from the   
 # center of self through the center of other.***

You should also define a function getCircle() that prompts the user for the appropriate information for a Circle, then returns that Circle object. This function is not a class method.