**Practice Problems for Objects And Classes \_ I**

1. Write a class definition as defined below:

Write a class definition for the Circle class.

The class has two instance variables, center and radius.

Note that the center is a tuple and contains two values - the x and y values. Use data mangling for naming your instance variables as discussed in class.

When the object is created, your code should check to make sure that the radius is strictly positive. If not, print an appropriate message.

In addition to the constructor, your class definition should also have the following methods:

1. A constructor to initialize all instance variables.
2. Accessor methods for all instance variables
3. Mutator methods for all instance variables
4. A \_\_str\_\_() method to print the center and radius of each circle object.
5. The definition should also have the following methods

a. get\_area(): This should compute and return the area of the circle

b. get\_circumference(): This should compute and return the circumference of the circle

c. grow(): Each time this method is called, the radius of the circle object will increase by 1

d. shrink(): Each time this method is called, the radius of the circle object will decrease by 1

1. Write a driver class to test the Circle class definition.

Create four Circle objects choosing appropriate values for the attributes. Then print the data for each circle. Generate a random number between 1 and 4: This will be the circle object you will select. Randomly select either 'shrink' or 'grow'. Then print out a message saying whether the circle was shrunk or grown and the center and the new radius. Apply this to the circle object from above. Repeat the above two lines four times. Print the four circle objects.

1. Rewrite the \_\_init\_\_ method from the above class by providing default values of (0,0) and 5 for the center and the radius respectively.
2. Assuming we desire to write code (to the extent possible in Python) that correctly follows the principles of encapsulation and abstraction, what is wrong with the following code?

**class Employee:**

**def \_\_init\_\_(self, name, id, dept, title):**

**self.name = emp\_name**

**self.id = emp\_id**

**self.dept = emp\_dept**

**self.title = emp\_title**

1. Assume that your class has an instance variable called **\_\_emp\_name.** What is wrong with the following code?

**def get\_emp\_name():**

**return \_\_emp\_name**

1. What is wrong with the following code?

**def \_\_str\_\_(self):**

**print(self.emp\_name, self.emp\_id, self.emp\_dept, self.emp\_title)**

1. Assume that you have correctly written a class definition for the Student class. The class contains the following instance variables: stu\_id, stu\_name, stu\_gpa. Also assume that this class definition follows all object oriented principles and is appropriately defined. What is wrong with the following code? (There are two errors)

**stu1 = Student(001, 'Carla', 3.85)**

**stu2 = Student(002, 'Mehdi', 4.0)**

**#Some more lines of code**

**print(stu1.stu\_id, stu1.stu\_name, stu1.stu\_gpa)**

**print(stu2.stu\_id, stu2.stu\_name, stu2.stu\_gpa)**

1. Assume that you wish to write a class definition for the Student class. The class contains the following instance variables: *stu\_id, stu\_name, stu\_gpa, stu\_advisor*. Note that *stu\_advisor* is an object of the Faculty class. This class has two instance variables: *fac\_name, fac\_title*. Assumimg that the Faculty class has already been correctly defined, write the \_\_init\_\_ method for the Student class using
   1. aggregation
   2. composition