CIS 1250 Python Program9 – Create a better class

# Turn in Requirements:

5 pts. Name your project LastnameP9, such as GarnerP9.

# Program Requirements:

1. 5 pts. Write the file name, your name, email address and purpose of the program at the top of your library source code in a comment.

# GarnerP7

# Programmer: Rob Garner

# EMail: Rgarner7 @cnm.edu

# Purpose: demonstrate how to define a class

1. 5 pts. Add comments as appropriate. Be sure that your program output is neatly presented to the user. Add documentation comments to your functions.
2. You are going to change the class you created in Program7 so that it has a constructor and some properties.
3. In your GeoPoint class make the following changes (note your variable, parameter and method names may be different. Adjust as needed.):
   1. Add a constructor \_\_init\_\_(self, lat=0, lon=0,description = ‘TBD’) that will initialize the class variables \_\_lat ,\_\_lon and the \_\_description. Notice that the constructor will also default lat and lon to zero and description to ‘TBD’ if they are not provided.
   2. Change the SetPoint method so that instead of individual coordinates SetPoint(self, lat, lon) it takes a single sequence.
   3. Add a property: Point = property(GetPoint,SetPoint). Make sure GetPoint and SetPoint are the names you used for the get and set methods you already wrote for points.
   4. Add another property: Description = property(GetDescription, SetDescription). Make sure GetDescription and SetDescription are the names you used for the get and set methods you already wrote.
   5. Change the distance method so that it takes a point instead of coordinates: Distance(self, toPoint) instead of Distance(self, lat, lon).
4. You will put your class in a separate library file. You will use an #include statement from the main part of your program.
5. In the main part of your program do the following:
   1. Include the class from the library file.
   2. Instantiate two points
   3. Instead of using SetPoint and SetDescription methods to set each of the points locations and descriptions do the following:
      1. Use the constructor to set point1 coordinates and description. It should look something like the following but with your own coordinates and description:

point1 = GeoPoint(12.3456,-123.4567,'Loc1')

* + 1. Use a constructor without any arguments to instantiate the second point and use it’s properties to set it’s values. It should look something like the following but with your own coordinates and description:

point2 = GeoPoint()

point2.PointCoords = 23.4567, -213.456

point2.PointDescription = 'Loc2'

* 1. Inside the “Do another (y/n)?” loop do the following:
     1. Ask the user for their location.
     2. Create a third point to represent the user’s location. You can use either the constructor or properties. With a constructor it will look like:

pointUser = GeoPoint(lat,lon, 'User Location')

* + 1. Use point1 and point2’s Distance method to find the distance from each point to the user’s location. Notice we are passing in the users point rather than the separate coordinates:

distanceToOne = point1.Distance(pointUser)

distanceToTwo = point2.Distance(pointUser)

* + 1. Tell the user which point they are closest to in this format:

You are closest to <description> which is located at <point’s x, y, z coordinates>

* + 1. Ask “Do another (y/n)?” and loop if they respond with ‘y’