NENG 685 Fall 2017
PF 4 Due Oct. 15, 2017

Name:

On Pre-flights:

- If you work with anyone else, document what you worked on together.
- If you are not using python, then substitute your language of choice when Python is specified.

Do not write in the table to the right.

Problem	Points	Score
1	10	
2	5	
3	5	
4	5	
5	5	
6	5	
Total:	35	

1. (a) (4 points) Describe the difference between class and object. Give an example of a class and an object (in the object-oriented programming sense).
   (b) (3 points) Define the three key features of object oriented programming.
   (c) (1 point) T/F: All variables, data types, and functions in python are classes.
   (d) (1 point) T/F: It is good coding practice to call dunder methods explicitly.
   (e) (1 point) What python built-in is available to show the attributes of an object?

Solution:

(a) A class defines the behaviors of a new kind of thing, while an object is a particular thing. I could have a dog class and an instance of that dog, and object, named Fred.

(b) Encapsulation is the property of owning data; Inheritance establishes a relationship hierarchy between models; Polymorphism allows for models to customize their own behavior even when they are based on other models

(c) True

(d) False

(e) dir()

2. (5 points) Create a class definition to describe an element. Give it attributes of density, atomic number, and atomic mass. Give it a constructor that requires the atomic number to instantiate the class and a method to print out the state of the object.

Solution:

```
class Element(object):
    def __init__(self, Z):
        self.density = 0.0
        self.A = 0.0
        self.Z = Z


    def print(self):
        print self.density, self.A, self.Z
```

3. (5 points) How can you overrule duck typing? Give your own example of why you might need to do so?

Solution: if isinstance():

If working in python 2.7 and you want to only do float division.

4. (5 points) Create a class definition to describe an isotope. Have the class inherit from the element class. Give it attributes of number of neutrons, and half-life. Give it a constructor that requires the number of neutrons and half-life to instantiate the class. Give it a method that can print the decay constant (note ensure the method is functional):

   $\lambda = \frac{ln(2)}{t_{1/2}}$

Solution:

```
from math import log


class Isotope(Element):
    def __init__(self, N, halfLife):
        self.N = N
        self.halfLife = halfLife


    def get_decay_const(self):
        print log(2)/self.halfLife
```

5. (5 points) What are class decorators? Why might you want to use them?

Solution: Class decorator we can add attributes or methods to an existing class. This can be useful if you want to modify the behavior of an existing class without modifying the source code.

6. (5 points) What is one concept that you found difficult in the reading?