NENG 685 Fall 2017
Pre-flight #2 Due Oct. 3, 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	5	
2	16	
3	3	
4	18	
5	5	
Total:	47	

1. (a) (2 points) Describe the difference between immutable and mutable data types. Give an example of each.

   (b) (3 points) Describe duck typing. Give an example of how this can be used.

Solution:

(a) Immutable data types cannot be changed, and there are no methods to do so. Examples are integers, floats, bools, tuples, and strings.

Mutable data types can be changes, and the data type contains the methods to do so. Examples include lists, arrays, sets, dictionaries.

(b) Duck typing allows for the variable to be used in the context that it is being applied rather than the underlying data type. Possible examples include:

1. Adding an int to a float
2. Indexing between strings, lists, dicts, etc

2. For each of the following, define how Python treats the data type and its key attributes, show how to define the collection of “a”, “b”, “c”, “b”, “a” as that data type, show how that data type is stored and output (i.e. if I define data=____, what would I get when I print data?), and show how to reference the string ”c” from that data type.

   (a) (4 points) list

   (b) (4 points) tuple

   (c) (4 points) set

   (d) (4 points) dictionary

Solution:

(a) Lists are pythons version of vectors (kind of). They are ordered, mutable, and can store different data types. myList = ['a', 'b', 'c', 'b', 'a']
['a', 'b', 'c', 'b', 'a']
myList[2]

(b) Tuples are immutable forms of lists. They are ordered and can store different data types. myTuple = ('a', 'b', 'c', 'b', 'a') = 'a', 'b', 'c', 'b', 'a'
('a', 'b', 'c', 'b', 'a')
myTuple(2)

(c) Sets are equivalent to mathematical sets. They are unordered, mutable, and can store different data types. mySet='a', 'b', 'c', 'b', 'a'
'a', 'b', 'c'
ref = mySet & 'c'

(d) Dictionaries a Python data structure that uses keys to access values. They are unordered, mutable, and can store different data types. myDict=1:'a', 2:'b', 3:'c', 4:'b', 5:'a'
1:'a', 2:'b', 3:'c', 4:'b', 5:'a'
myDict[3]

3. (3 points) Python has 3 major types of flow control and logic. Name them.

Solution: Conditionals, Exceptions, and Loops

4. (a) (3 points) Write Python code to print "Bazinga!" if the name is "Sheldon", "I fooled you!" if the name is not equal to Sheldon.
   (b) (3 points) Write Python code to handle a TypeError for setting *ans* equal to 1 + "Three".
   (c) (3 points) Write Python code to raise a TypeError exception if the type of a variable is str.
   (d) (3 points) Write Python code to continue to calculate $f_i(x)$ while $f_i(x) - f_{i-1}(x)$ is greater than some convergence tolerance.
   (e) (3 points) Write Python code to calculate $\sum_{i=0}^{100} i$ using a for loop.
   (f) (3 points) Use list comprehension to create another list containing all words that contain the letter 'e' from the list of ['Bryan', 'Robert', 'Amy', 'Nick', 'Kevin'].

Solution:

(a)
```
if name == 'Sheldon':
    print 'Bazinga'
else:
    print 'I fooled you!'
```

(b)
```
try:
    ans=1+'Three'
```

```
    except:
        TypeError 'unsupported operand type(s) for +: int and '"string"'
```

(c)
```
if type(var) == str:
    raise TypeError
```

(d)
```
while delta > conTol:
    prevVal = curVal
    curVal = f(x)
    delta = curVal - prevVal
```

(e)
```
sum = 0
for i in range (0, 101):
    sum += i
```

(f)
```
names = ['Bryan', 'Robert', 'Amy', 'Nick', 'Kevin']
concat= [name for name in names if 'e' in name]
```

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