NENG 685 Fall 2017
PF 3 Due Oct. 10, 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.

Run LaTeX again to produce the table

1. (10 points) Define a function that returns the value of up to a 4th order polynomial for a given input and set of parameters for the polynomial. Make the linear terms required inputs to the function; make the other terms optional inputs.

Solution:

```
def poly(x, y0, x1, x2=0.0, x3=0.0, x4=0.0):
    return y0+x1*x+x2*x+x3*x+x4*x
```

2. (5 points) Define a function that returns the product (i.e. Π) of an unknown set of numbers.

Solution:

```
def prod(*args):
    ans=args[0]
     for p in args[1:]:
        ans = ans*p
    return ans
```

3. (a) (2 points) What does the '*' operator do as a prefix?
   (b) (2 points) What does the '**' operator do as a prefix?
   (c) (3 points) What is the difference between args and kwargs?
   (d) (3 points) Why might you use kwargs instead of args? Give an example.

Solution:

(a) Unpack a sequence

(b) Unpack a dictionary/mapping

(c) ** is used to prefix the name, and kwargs are keyword, valued paired.

(d) Some reasonable responses:

- To pass keyword valued optional arguments to a sub-function
- To add robust "super-user" behavior without cluttering up the API.

4. (a) (1 point) What properties do first class objects have in Python?

   (b) (3 points) In your own words, describe the meaning of global scope, local scope, and module scope.

   (c) (6 points) What is the value of 'var' at each print statement in this code snippet?

```
var = 12
print var

def func(var):
    print var
    var = 6
    print var
    def funct(var):
        print var
        var = 9
        print var
    funct(var)
func(var)
print var
```

Solution:

(a) 1. They may be dynamically renamed, like any other object.
    2. Function definitions may be nested inside of other function bodies.

(b) Local scope = variables defined inside of a function have lifetimes that end when the function returns.
Global = module = The function may access and modify these variables, so long as their names are not overridden by local variables with the same names.

(c)

```
var = 12
print var 12

def func(var):
    print var 12
    var = 6
    print var 6
    def funct(var):
        print var 6
        var = 9
        print var 9
    funct(var)
func(var)
```

print var **12**

5. (5 points) What is recursion in Python? Why does Python implement default limits for recursion?

Solution: Recursion allows a function to call itself. Without a stop condition, this will create an infinite loop, so Python sets a recursion limit.

6. (5 points) Write a lambda function named sum to calculate the sum of a + b.

Solution: sum = lambda x,y: x+y

7. (5 points) In your own words, describe generators. Thinking of the topics covered in this class, when might this be useful?

Solution: Generators allow for periodic outputs and access using the next() built-in. An example use would be a random number generator.

8. (5 points) What is one concept that you found difficult in the reading?