Lab 11

Learning Objectives:

I. **What characters do I type next?** *How do I use objects oriented programming concepts in Python?* In this lab, you will be using object-oriented programming techniques in Python and will get experience writing classes, along with importing and inheriting.

E. **Don't Reinvent the Wheel** *Don't write code that you don't need to.* Object oriented programming approach allows you to write very reusable code to reduce the amount of separate programs you need to write. In addition you will be using Python built-in objects and functions to internally store data in your class.

Part 1 will help you put all the different parts of a list comprehension together, one by one.
Part 2 will have you work with small examples showcasing concepts from functional programming.

Part 1: Building a List Comprehension from the Ground Up

Comprehensions have a lot of pieces of syntax. This part will help you use them all together, implementing one at a time. When you're done, save your solutions in a file named `list_comp.py`.

Write a list comprehension that:

- **Returns a list of the first 10 letters of the alphabet.** Hint: here's an alphabet: (import the string module first) `string.ascii_lowercase`.
- Returns a list of the first 10 letters of the alphabet **except for the sixth letter.**
- Returns a list of the first 10 letters of the alphabet, except the sixth one, **each repeated 1, 2, and 3 times:** `['a', 'aa', 'aaa', 'b', ..., 'c', ...]`
- Returns the list like the above, but in a grid: `[['a', 'aa', 'aaa'], ['b', ...], ['c', ...], ...]]`
- Returns a list like the above, but if the number matches the index of the character *mod 3* (e.g. `'c'` and `3`, instead print a single capitalized version of that character: `[['A', 'aa', 'aaa'], ['B', 'bb', 'bbb'], ['C', 'cc', 'ccc'], ['D',...]]`

Part 2: Function-fu

This part of the lab is designed to be done interactively, using an IPython Jupyter Notebook and the NumPy library functions. You are now familiar with nearly all the major features in Python, and now is your chance to see how powerful the language can be. Your task is to write functions for the following tasks, in as compact and elegant a form as possible. When you're done, your solutions should be all written in a IPython notebook named `functions.ipynb`.

- Squares: Create a list of the first 10 square numbers. First use a list comprehension, then use a `lambda`

function and `map`.

- Products: Use a `lambda` function with `reduce` to multiply up the numbers from 1 to 5 (use `from functools import reduce`).
- Filenames: Say that you have a string with a list of file names `"test1.py test2.py test3.py ..."` and you want to output a list of just the filenames without the .py extension. Do this with a list comprehension.
- Filenames bonus: See if you can only return the filenames of .py files and ignore files with other extensions.