ICS 313 Homework 6

Introduction to Python

0. Goals and Overview

The goals of this assignment are to have you practice using Python and to have you explore a small part of Python's libraries.

1. random program

Write a python program that does different things depending on its command-line arguments (see also here):

- when called with no arguments, your program prints a random number between 0 and 100, inclusive.
- when called with one numeric argument > 0, your program prints a random number between 0 and that number, inclusive
- when called with two numeric arguments n and m, with n < m, your program prints a random number between (and including) n and m
- when called with argument -s followed by a numeric argument n > 0, your program prints a random lowercase string of length n
- if one of the arguments is -c followed by an numeric argument n > 0, and the rest of the arguments follow one of the preceding patterns, your program prints n random numbers or n random strings
- when called with argument -p followed by one or more arguments, your program prints a random permutation of the arguments that follow -p
- if the arguments don't satisfy any of the above, your program should print a suitable error message and quit

Examples:

```
$ random
43
$ random 1
0
$ random 1
0
$ random 1
1
$ random 1 10000
750
$ random 5000 10000
9093
$ random -s 10
```

```
bnyovuqlbw
$ random -c 3 -s 5
btuwl
kymwg
dhfkf
$ random -p hello world I am a random program
a
random
I
program
world
hello
am
```

2. Command line and arguments

If you are on a unix computer (which includes any linux or Apple computer), use the provided terminal or shell to test your python script. If you do not have your own unix computer, you are encouraged to use ssh to access the shell on uhunix.hawaii.edu. Uhunix has both python version 2, simply called `python`, and python version 3, `python3`. As for any new python program, I encourage you to use python3.

If you wish to also run your program in a Windows command window, you should be able to do so, but I am unable to provide exact directions. There seems to be lots of information on the web.

3. Testing

In general, careful mathematical analysis is needed to determine whether a random number generator is producing pseudo-random numbers that satisfy a desired distribution.

For this assignment, we will assume that the Python random library provides values of a suitable distribution. However, make sure you are using the library correctly. As one test, repeatedly run

```
random -c 1000 1 | fgrep -c 1
```

If your random number generator is correct, most of the time this should print a value between 400 and 600.

You are of course encouraged to also devise your own tests.

4. Plotting

Write another python program to get from your random program 1,000 random numbers, each between 0 and 100, and to track how many occurrences you get of each of the 101 possible values. Your python program for this section must then plot these numbers as a bar graph.

Please review figure 14.6 and the python part of section 14.2.4 in the textbook if you need help figuring out how to get the output of one program into a different program.