Learning Python

Francisco Iacobelli

1 Before you begin

- Download the latest python (3.x). As a suggestion, download Anaconda at `https://www.anaconda.com/`. Anaconda installs the latest python version and I recommend you develop with Visual Studio Code or Spyder. PyCharm is also ok.
- Teach yourself python. You can start by checking this youtube playlist with python for jave programmers.
- Learn how to interact with python from the command line, not from your IDE.This link contains the answer
- Learn how to read parameters from the command line in python. This may help

2 TASK

- Create a program that is invoked with a list of integers from the command line. The program should output the integer that is repeated the most times. If there is a tie, chose any number among those that tie. The program should be called `most_times.py` and should be run from the command line like so:

  ```
  python most_times.py n1 n2 n3 ...
  ```

  where `n1 n2...` are integers. For example:

  ```
  python most_times.py 12 1 3 12 5
  ```

  should output `12` because it is the number that occurs the most times.

- Create a program that reads a filename from the command line and prints the number of unique words, total number of words (not unique), characters (including spaces) and lines in that file. It must be called wc.py and must be run like so:

  ```
  python wc.py filename
  ```

  where `filename` is the actual name of the file to read.

  As an example, download this sample file: heaalthyeating.txt'.

  Then, when your program is run with this file (`python wc.py healthyeating.txt`), the output of the program should be in the ballpark of `lines:42, unique:429, words:819, chars:4878`. The numbers may vary a bit, but not a lot.

- Modify your program to print the most common and least common word. You must use dictionaries for this. Using the `healthyeating.txt` file, `python wc.py healthyeating.txt` will print:

  ```
  lines:42, unique:429, words:819, chars:4878
  Most frequent word:and (33 times), Less frequent word:People (1 times).
  ```

  Again, your output may be different than mine if you treat words differently (e.g. you turn all words to lowercase or something like that).

- Create a file named `basic_stats.py` that has the following functions:
  - `avg`: takes a list of numbers as a parameter and returns the average.
  - `l_sqr`: takes a list of numbers as a parameter and returns a list with the corresponding values squared.
  - `var`: takes a list of numbers X and computes

    $$\frac{\sum_{x \in X}(x-\mu)^2}{n}$$

    where μ is the average of the numbers in X and n is the number of elements of X.

*Note: functions must **return** values, not print them.

To test your file, you may type the following on a python shell:

```
from basic_stats import *

print(avg([23,32,43,54])==38.0)
print(sum(l_sqr([23,32,43,54]))==6318)
print(var([23,32,43,54])==135.5)
```

And, when you run it, you should see three `True` outputs.

3 CRITERIA

- The only library you are allowed to use is `sys`.
- If your work is submitted as specified (check the next section): 30pts.
- If the programs do not have runtime errors running the tests: 20pts.
- each program running correctly is 10 points, and:
  - `basic_stats` must not print anything. Functions should be named exactly as specified and should return values (5pt.)
  - If `wc` uses dictionaries (5pt.)
  - You will get an extra point for every function in `basic_sats` that only consists of a return statement that performs the correct functionality (i.e one liners).

4 SUBMIT

One ZIP (not arj, rar, etc.) file with the three programs (`most_times.py`,`basic_stats.py` and `wc.py`. All names of programs and functions should be exactly as specified above, including underscores and case.