Joanna Klukowska
joannakl@cs.nyu.edu

Assignment 9
Due Date for Programs: November 30, 2016

☞ The parts of this assignment marked with the ”alone” icon, , should be completed individually by each student.

You may discuss any parts of the assignments with your classmates and tutors (or anyone else) but you are responsible for understanding all of the parts of the assignment that you submit. Any sharing or copying of the parts of the assignments marked with ”alone” icon, , will be considered cheating.

You should not use any features of Python that have not been covered in class or recitations. If you have doubt if you are allowed to use certain structures, just ask one of the instructors.

Problem 1 (25 points): Stats on strings

Write a program that prompts the user for a phrase (a string) and computes some info about the characters contained in the string.

The program should print the information as shown in the samples below. If the particular type of character is not present in the string, its count should be shown as zero.

Here are a few sample runs of the program:

Output:

```
Enter your phrase: Hello. Today is November 14, 2016.

Length:  35
Spaces:  6
Vowles:  9
Consonants:  11
Digits:  6
Non-digits:  29
Letters:  20
Non-Letters:  15
```

Output:

```
Enter your phrase: Flying elphants weigh only 500lb. each! Isn't this amazing?

Length:  59
Spaces:  8
Vowles:  14
Consonants:  30
Digits:  3
Non-digits:  56
Letters:  44
Non-Letters:  15
```

Output:

```
Enter your phrase: I do not know what to enter.

Length:  28
Spaces:  6
```

Joanna Klukowska
joannakl@cs.nyu.edu

```
Vowles:  7
Consonants:  14
Digits:  0
Non-digits:  28
Letters:  21
Non-Letters:  7
```

HINT: this program needs to iterate over the user provided string and increment counters based on return values of some of the methods for the strings (like `isalpha()`, `isspace()`, etc. - see the lecture slides).

Note: if the user does not enter and just hits Enter key, the program should display all zeros:

Output:

```
Enter your phrase:

Length:  0
Spaces:  0
Vowles:  0
Consonants:  0
Digits:  0
Non-digits:  0
Letters:  0
Non-Letters:  0
```

Comment your source code by 1) briefly describing parts of your program 2) including your name, the date, and your class section at top of your file (above everything else) 3) **documenting all the functions following the IPO format**

What to Submit

This program should be named (i.e., the name of the file containing the program should be) `string_stats.py`. You only need to submit the source code for this problem.

Problem 2 (25 points): Counting Words

Write a program that counts the number of spaces (and hence the number of words) in the user provided phrase. We will use this idea for a future program that works with larger amount of text.

The program should prompt the user for a phrase and then calculate the number of spaces in that phrase. Assuming that there are no spaces before the first word, no spaces after the last word and only one space between all other words, the number of words should be one more than the number of spaces.

HINT: you may find the slicing operator and the `find()` method handy for this problem.

Here is a sample run of the program:

Output:

```
Enter your phrase: Hello. Today is November 14, 2016.
Found  5 spaces
Should have  6 words
```

Output:

```
Enter your phrase: Flying elphants weigh only 500lb. each! Isn't this amazing?
Found  8 spaces
Should have  9 words
```

Output:

```
Enter your phrase: I do not know what to enter.
Found  6 spaces
Should have  7 words
```

But if the assumptions about spaces are wrong, the count of words will be off.

Output:

```
Enter your phrase: I like   to   use   a    lot   of   spaces in my      phrases.
Found  30 spaces
Should have  31 words
```

Comment your source code by 1) briefly describing parts of your program 2) including your name, the date, and your class section at top of your file (above everything else) 3) **documenting all the functions following the IPO format**

What to Submit
This program should be named (i.e., the name of the file containing the program should be) `word_count.py`. You only need to submit the source code for this problem.

Problem 3 (50 points): Popular Vowels in Literature

This problem is your mini-project. You need to do a bit more than just write the code, although the code writing is probably going to be the most challenging part.

Your task is to pick six (or more) books and analyze the vowel-use in each of them. Make your book selection follow some theme that is interesting for you For example, ...

- ...you may pick books in two different languages. The results may demonstrate different vowel statistics between the two languages but similar ones for books in the same language.
- ...you may pick books representing two different genre and see how the vowel use compare there.
- ... you may pick books from two different authors or from two different time periods and see if the vowel use differs there.
- ...

The program should describe your ”theme” in the header of the file (where you are describing the program itself).

Visit Project Gutenberg website, https://www.gutenberg.org/ to select the books. You should download the `Plain Text UTF-8` format of the books. You will need to upload the text files for this book with your assignment so we can run your programs.

The program should count occurences of each of the vowels (treat ’y’ as a vowel for this assignment). The results should follow the format below, although if you chose to display any additional interesting info, that’s fine as well. At the minimum the program should display the following for each book:

- book title
- counts of each vowel per hundred letters (calculate the total number of a given vowel, divide it by the number of total letters and then multiply by 100) [WARNING: the total number of letters is not the same as the total length of the string containing the text of the book - that value includes a lot of punctuation, spaces, possibly digits, etc.]
- a histogram that follows each count (the number of bars in the histogram should correspond to the value rounded up to the nearest whole number)
- total number of vowels per 100 letters

Given the number of repeated tasks (the program has to repeat each calculation at least six times - once for each book), you HAVE to use functions for everything. There are no specific functions that you are required to write but try to wrap every task in the function. Even printing of the results should be done by a function taking parameters so you do not have to repeat all the code required to format and align the results.

Majority of the calculations required for this project are done by processing a HUGE string that contains the text of the entire book. To open a text file that contains the book you need the following lines of code:

```
# open file for reading
myfile = open(file_name, "r")

# read in all data as one long string
all_text = myfile.read()
```

The first line opens the actual file so that the Python program can read it. The second line reads the entiere text is places it into one HUGE string variable (in the above case the string variable is called **`all_text`**, but you can call it whatever you want). For example, if I saved the Hamlet file using the name **`Hamlet_WilliamShakespeare.txt`**, to read the book from my Python program I would use:

```
# open file for reading
myfile = open("Hamlet_WilliamShakespeare.txt", "r")

# read in all data as one long string
all_text = myfile.read()
```

Once that's done, I can use the variable **`all_text`** to access the text of the entire book.

NOTE that this assumes that the book files are in the same directory as the code itself. Make sure that you save the books that you download in your source code directory for this lab.

Here a run of my program with six different books (there is not much of a theme here). Your programs should not use the same books as the ones below (although you may want to downlaod those as well to verify the numbers produced by your program).

Output:

```

---------------------------------------------------
---------------------------------------------------
Alice's Adventures in Wonderland - vowel statistics
---------------------------------------------------
---------------------------------------------------

Counts per 100 letters:
  A:   7.57  ||||||||
  E:  12.43  ||||||||||||
  I:   6.30  |||||||
  O:   7.40  ||||||||
  U:   3.16  ||||
  Y:   1.99  ||

Total vowel count per 100 letters::  38.84


-------------------------
-------------------------
Hamlet - vowel statistics
-------------------------
-------------------------

Counts per 100 letters:
  A:   7.09  ||||||||
  E:  11.38  |||||||||||
  I:   5.92  ||||||
  O:   8.32  |||||||||
  U:   3.27  ||||
  Y:   2.39  |||

Total vowel count per 100 letters::  38.38


----------------------------------------
----------------------------------------
Sense and Sensibility - vowel statistics
----------------------------------------
----------------------------------------

Counts per 100 letters:
  A:   7.55  ||||||||
  E:  12.41  |||||||||||||
  I:   6.50  |||||||
  O:   7.96  ||||||||
  U:   2.80  |||
  Y:   2.16  |||

Total vowel count per 100 letters::  39.39


------------------------
------------------------
Faust - vowel statistics
------------------------
------------------------

Counts per 100 letters:
  A:   4.40  |||||
  E:  13.79  ||||||||||||||
  I:   7.54  ||||||||
  O:   2.48  |||
  U:   3.46  ||||
  Y:   0.19  |

Total vowel count per 100 letters::  31.86


------------------------------
------------------------------
Pan Tadeusz - vowel statistics
------------------------------
------------------------------

Counts per 100 letters:
  A:   8.50  |||||||||
  E:   7.07  ||||||||
  I:   8.21  |||||||||
  O:   6.71  |||||||
  U:   2.29  |||
  Y:   3.64  ||||

Total vowel count per 100 letters::  36.41


----------------------------------------------
----------------------------------------------
La Divina Commedia Di Dante - vowel statistics
----------------------------------------------
----------------------------------------------

Counts per 100 letters:
  A:  10.28  |||||||||||
  E:  11.39  ||||||||||||
  I:   9.59  ||||||||||
  O:   9.23  ||||||||||
  U:   3.31  ||||
  Y:   0.08  |

Total vowel count per 100 letters::  43.88
```

Joanna Klukowska
joannakl@cs.nyu.edu

Comment your source code by 1) briefly describing parts of your program 2) including your name, the date, and your class section at top of your file (above everything else) 3) **documenting all the functions following the IPO format**

What to Submit

This program should be named (i.e., the name of the file containing the program should be) `popular_vowels.py`. **You need to submit the source code as well as all the book text files for this problem.**

Grading

The only way to receive the credit for the worksheet problems is to hand them in before the end of the lab session in which they are given.

The programs are graded based on correctness, style of code, design and documentation.

What and how to submit?

You should submit the source code file for each program to NYU Classes by the due date stated above. Make sure that you get an email confirmation after you submit the assignment. You should keep that email until the grades are returned - it is your proof that the assignment was submitted! If you do not get an email confirmation, you should try to resubmit the assignment. If you do not get that email, it means that we did not get your assignment.