ISTA 130: Spring 2020
Programming Assignment 7
(100 points)
String Manipulation, File Reading & Writing

Due: April 16th at 11:59 pm

(submit via D2L Assignments)

Please read the instructions below carefully and follow them closely. All problems (and sub-parts of the problems) are required except as noted in the instructions below. If you have any questions, please email the instructor or one of the section leaders.

Important: **You will lose 5 points if you use the tab character for indentation.**

Important: Your filenames must be identical to the filenames given below. For any functions you are asked to write, the function signature (header) must be exactly as described in the instructions. That is, you must use the exact function names given in the instructions, you must have the parameters the instructions ask for, and the parameters must be in the order the instructions give.

Important: Make sure you always save a backup of your work somewhere safe (such as UABox, Dropbox, etc).

Important: You are not just graded on whether your code produces the same result as the examples show. You should be using what you've learned to do your best to write good code. For example, things like poorly named variables/functions, duplicating code where you could instead use a loop, and missing documentation will cost you points.

1.0) MAD LIBS (100 POINTS)

A **Mad Lib** is a story template that has many missing words, each indicated by a label giving the category (**part of speech**) of the missing word. For example, a story template will contain the word **`NOUN`** in many places instead of specific nouns, the word **`ADJECTIVE`** in many places instead of specific adjectives, etc. Before reading the story, the reader asks listeners to provide the missing words without indicating where and how the words will be used in the story.

If, for example the story template is:

```
Once upon a NOUN, three PLURAL NOUN VERB PAST in the NOUN.
```

The reader would first ask “Give me a noun” to which the listener might reply “piano”. The reader would then ask “Give me a plural noun” and the listener might say “hamburgers”. The reader would next ask “Give me a past tense verb” and the listener might say “frolicked”. Finally, the reader would ask “Give me a noun” and the listener might say “dumpster”.

The reader replaces the missing words with those provided by the listener and reads the result:

```
Once upon a piano, three hamburgers frolicked in the dumpster.
```

At this point everyone in the room giggles until their eyes water.

For this assignment you will write a program that reads a Mad Lib story template from a file (template files will be provided to you).

The Mad Lib template files may contain any of the following part of speech labels:

- **NOUN**
- **PLURAL NOUN**
- **VERB**
- **VERB PAST**
- **ADJECTIVE**

The program asks the user to supply each of the needed words and writes out the completed version of the story to a new file (Note: the label **VERB** on its own indicates present tense). You may assume that the two part of speech labels with two words (**PLURAL NOUN** and **VERB PAST**) will never be split over two lines in the file. The program will also print a simple numerical analysis of the contents of the file (explained below).

1.1) DETAILS

Create a new file called `madlib.py`. In the file:

1) Write a function called **`print_report`** that takes a single string parameter for the name of a file.

The function will:

i. Read the given file and print a report giving the following information:

a) the total number of vowels in the file (for this assignment `y` is not a vowel)

b) the total number of consonants in the file

c) the total number of white spaces in the file (spaces, tabs, newlines)

d) the total number of punctuation characters in the file (for this assignment assume that punctuation is anything that is neither a letter from the alphabet nor a white space)

e) the total number of characters in the file

f) the percent of the file composed of vowels, consonants, white spaces, and punctuation characters.

g) begin and end with a blank line.

ii. The function should print the report in a pretty table similar to the one shown in the following example.

```
-------tortoise.txt------
Vowels:                232
Consonants:            368
Whitespace:            150
Punctuation:            22
------------------------
Total:                 772

Percent vowels:       30.1
Percent consonants:   47.7
Percent spaces:       19.4
Percent punctuation:   2.8
========================
```

In order to create a table that looks like this one, you must have the file name centered in a line of dashes, 25 total characters, the alphabetical text should be left justified in a field of length 20 and the numbers (as strings) should be right justified in a field of length 5 (see the times table example from class or google `ljust` and `rjust`). The following link should prove useful:

https://docs.python.org/3/library/stdtypes.html#string-methods

- For the percents, you'll need to use the built in **`round`** function. Here's an example:

```
round(3.14159265, 4) --> 3.1416
```

- Experiment with it in a shell until you know how to use it.
- Hint: it might be a good idea to create your own text file to test this code on. For example, you might first try it with a file that has exactly 5 vowels in it and nothing else...
- Test your function and verify it works correctly before moving on.
- **REMEMBER THAT SPACES AREN'T THE ONLY FORM OF WHITESPACE**

2) Write a function called **`replace_parts_of_speech`** that takes two parameters. The first is a string representing a line from a file. It may contain part of speech labels that need to be replaced by words (e.g. `"The ADJECTIVE NOUN in the NOUN VERB PAST."`). The second is a string indicating which part of speech label to replace, e.g. `"NOUN"`.

i. For each occurrence of the given part of speech in the given string ask the user for a word of the appropriate type.

ii. Replace the part of speech label with the word provided by the user

iii. After replacing all parts of speech of the indicated type, return the new version of the given string

- For example, calling the function like this:

```
replace_parts_of_speech("the NOUN VERB PAST the NOUN", "NOUN")
```

would cause the function to ask the user to enter a noun twice.

Assuming the user entered `"dog"` for the first noun and `"duck"` for the second, the function would return:

```
"the dog VERB PAST the duck"
```

Note: When asking the user for words, your prompts should like these examples:

```
Enter noun:
Enter plural noun:
Enter verb:
Enter verb past:
Enter adjective:
```

Test your function and verify it works correctly before moving on.

3) Write a function called **`complete_mad_lib`** that takes a string parameter for the name of a Mad Lib template file to read. The function will:

i. Read the indicated template file line by line and

- Replace all part of speech labels with words entered by the user
  - you'll need to figure out how to use the function you wrote to accomplish this
  - Do it in this order: **PLURAL NOUN**, **VERB PAST**, **VERB**, **NOUN**, **ADJECTIVE**

ii. Write the completed story out to a new file:

- the new file should have the same name as the original file, prepended with the string `"MAD_"`)

(e.g. if the original file was named `"tortoise.txt"` the new file should be `"MAD_tortoise.txt"`)

Test your function and verify it works correctly before moving on.

4) In `main` do the following:

i. Ask the user to enter the name of a Mad-Lib template file (assume the file is in the same directory as this program)

ii. Call your function to print the character report on that file

iii. Call your function to have the user complete the Mad Lib story

5) Verify that your documentation makes sense and that you've added documentation to each of your functions.

6) **Verify that your program works**

7) Upload your file to the Hw7 Assignments folder on D2L