CDS-230-001 - FALL 2023

PROBLEM SET 4

9/28/23

Instructions: Use the PS template provided in class to enter your name and your answers. **When you are done run your script and make sure it executes without syntax errors**.

Exercise 4.1: Average of random numbers (10 points)

a) Write a function, call it *my_avg*, that takes in a list of numbers and returns the average of the numbers. The function should compute the average by using the definition of "average". (Remember: average = sum of values divided by number of values).

b) Use Python's intrinsic function *randint* (see e.g. PS2) to generate 1000 random integers between -10 and 10

c) Use *my_avg* to compute and print the average of the random numbers in (b). As always, print the results using an informative print statement.

Does the value you get in (c) make sense to you? Explain your answer.

Exercise 4.2: Finding a sequence in a file (10 points)

The next question uses a text file named *war_and_peace.txt* which is in the data directory of the course website (see under Course Resources→Data). Download the file, open it with an editor and look at its contents.

In this exercise your task is to write a function named *find_sequence* that will create a **list of all of the locations** of a given string sequence. The arguments (input) to this function are the filename (*war_and_peace.txt*) and a string denoting the **sequence** to be found. The output of the function should be a list containing all the sequence **locations**. This will be a large list, so do not print it!

Test your function with *sequence='you know'* and *sequence = 'I am not'* and **print the length** of the returned lists (ans. 147 and 40 respectively).

Note: You must read the file in the function *find_sequence*
Hint: You need to use the string method *find()*

Exercise 4.3: File IO: counting words (10 points)

You will continue working with the file *war_and_peace.txt*.
Your task is to

a) Open the file *war_and_peace.txt* and read its contents into a **list**. Then, answer the following questions (in all cases use an informative print statement):

b) How many total lines are there in the file? Write code and print result. (Ans. 66032)

c) How many *non-empty*[1] lines are there in the file? Print result. (Ans. 52175)

d) How many times does the word **Hippolyte** appear? Print result. (Ans. 51)

Exercise 4.4: File IO: DNA sequence (20 points)

Each cell in any animal or plant contains a vast amount of DNA, a complex molecule consisting of two strands linked together with four *nucleotides* - the building blocks of DNA - of which there are four variations : **A,C,G,T**. A single human cell has approximately 3 billion nucleotides which make up the human genome.

In this exercise we will count ”letters” in a DNA sequence stored in the file *dna.txt* which is in the data directory of the course website (see under Course Resources→Data). Download the file, open it with an editor and look at its contents. You should see that it contains letters (a, c, g, t) that represent the four types of bases found in a DNA molecule.

Your task is to first open the file *dna.txt* and store its contents into a string called *data*. You should write a function that takes in a filename parameter, reads the file, and returns a string with the contents of the file. Then

a) Use len() to compute the length of the string, store the value in variable **data_len**.

b) Use the string count() method to count the number of occurrences of each a, c, g, t. Store in variables **a,c,g,t** respectively.

c) Calculate the difference between **data_len** and the sum of all the **a,c,g,t** nucleotides calculated in (2). The difference should be zero but it’s not. That means that there are letters in the sequence that are not in the set **a,c,g,t**. These type of ”errors” are common in data and must be accounted for.

d) Use set() to determine all the unique letters in the sequence.

e) Use set operations to get the letters that are not nucleotides (the ”errors”).

f) Use a for loop and a dictionary to count how many letters there are of each both nucleotides and ”errors”.

[1]Note: An empty line has no text, it’s just a newline character, ’\n’