Assignment 5

Objectives: Practice working with strings and using files. Practice the Pythonic way to make `for` loops.

Note: Include DocStrings in each script you submit analogous to the following:

```
"""This script characterizes squirrel-human interactions in NY.


Submitted by Mauricio Arias, NetID ma6918
[Include a short explanation here of how the script does that.]
"""
```

Part 1. Searching in text files.

Task 1.1. Squirrel!

In the attached file about 3000 interactions with squirrels were documented by a group of volunteers. Use that file to test the script described below.

Write a script that requests a string and a text file name. The script opens the file and counts how many lines contain the string provided and counts how many times in total the string occurs in the file. It closes the file and reports the results. The script then goes back to asking user input again. (Notice that a single line can contain more than one occurrence of the string.) Include a results file made manually with the following data:

1- How many lines contain the word adult? How many lines contain the word juvenile? How many lines contain neither?

2- What are the line counts for the fur colors (as long as it is in the line it counts): cinnamon, gray, black, brown and white.

Save your answers as a simple text file (.txt) according to the instructions below.

Submit your script as `[NetID]_characterizing_squirrels.py`. Submit your answers as `[NetID]_squirrels_chars.txt`. (2.5 pts)

Task 1.2. Taxi!

In the attached files for-hire-vehicle trips are listed in NYC. For the second week of May 2020 from Sun May 10th to Sat May 16th, count how many trips occurred: a trip counts if it either started or ended in that period or both. Do the same for the second week of May 2021 from May 9th to May 15th. Your script should ask for the first file to use and the first day of the week. It should do the same for the second file and day. (Assume it is always the second week so running over to the next month does not occur.) It then checks the files and prints a table comparing those two weeks, each value should be separated by a comma. Don't use spaces. For example,

```
Sunday,22,66
Monday,10,45
```

etc.

Copy and paste the output in a text file and name it as instructed below.

Submit your script as `[NetID]_counting_trips.py` and the results as a csv file `[NetID]_taxi_results.csv`. (2.5 pts)

Part 2. Concatenating and playing with Strings.
DNA: the instructions
In all organisms known, long molecules of DNA are present and they are carefully maintained. These molecules are composed of two anti-parallel strands containing combinations of the building blocks denoted by the letters A, C, G and T: they are called DNA bases. These long molecules contain information for how to make the different proteins the cell uses among other things.

Technologies have been developed that allow us to read the full sequence of these molecules which can extend for hundreds of millions of bases. The information in the two strands is almost completely redundant. Therefore, when these sequences are reported only the sequence for one strand is given: the sequence for the other strand can be easily deduced but we won't go into that today.

A common type of format used for reporting sequences is the FASTA format: pronounced FAST-A. In this file a line has the description of the sequence. The following lines contain the sequence information separated into many lines: a common practice is to make each line 70 bases or shorter. However, the sequence is continuous.

Task 2.1. In many types of bacteria, palindromic sequences of 6 nucleotides play important roles in protecting the bacteria from viruses that target it. This defense mechanism is called restriction-modification. Unfortunately we won't go into the details here. However, we will gather some information about the abundance of these sequences in the genome of a virus for *Escherichia coli*, the lambda phage.

A genomic sequence is considered palindromic if it follows the format XYZZ'Y'X'. The primed letters represent what is called the complement: for A, the complement is T and viceversa; for C, the complement is G and viceversa. Therefore, AAATTT is palindromic according to this definition. ACTAGT is also palindromic. Make a function that provides the complement for any base. Use this function to make a function that provides the complement for any string of bases and writes it in reverse order: this is called the reverse complement. In this way any 6-base palindrome is any combination of 3 bases followed by the corresponding reverse complement: therefore, there are only 64 possible palindromes. Notice that the code `for base in "ACGT":` loops through all the possible bases in order. Use nested `for` loops to generate all 6-base palindromes and for each check in the genome of the lambda phage how many times it occurs. Open a file called `[NetID]_lambda_palindromes.csv`, and write the results as comma separated values: one palindrome per line. To go back to the beginning of a file use the seek() function. It is also useful to go back to a specific position in the file, not necessarily the beginning. (Research the seek() function on your own. Use the book or the internet for that. Make sure your source is reliable.) Repeat the analysis for *E. coli*.

Note: Even though most of the reported genomes are in uppercase format, some parts of the genomes are written with lowercase letters. Make sure you don't miss palindromes in those regions.

Submit your script as `[NetID]_palindromes.py`. Don't forget to submit your `[NetID]_lambda_palindromes.csv` and `[NetID]_Ecoli_palindromes.csv` files. (2.5 pts)

Task 2.2. In the genomes of many organisms low complexity regions are abundant. These regions have by definition very simple composition, in many cases being only repetitions of a small sequence. Some times they are long stretches of a single base: which base might have important implications. (Long stretches of A or T might indicate some functional and dangerous sequences might be nearby: if you are interested you can look up reverse transcription of viruses and integration. It is a fascinating topic.) Generate a script that identifies the longest stretch of each base (A, C, G or T) in the human mitochondrial genome. Your script should report the length of this stretch, how many times it occurred and how many times the length exceeded 10 bases. Make your script general by requiring that the user inputs the file to be processed. (See the notes about the FASTA format in the preceding section.) Ask for the name for the output file in your script. Save your results in the file `[NetID]_mitochondrial_stretches.csv`. Repeat the analysis for chromosome 21. Save your results in the file `[NetID]_chromosome21_stretches.csv`. Note: some of the genomes have long stretches of N. They simply represent places where the actual sequence has not been obtained satisfactorily.

Submit your script as `[NetID]_stretches.py`. Don't forget to submit your `[NetID]_mitochondrial_stretches.csv` and `[NetID]_chromosome21_stretches.csv` files. (2.5 pts)