Homework 5

In this homework, you have been given some information about all of the movies that have been nominated for Best Picture in this year's Academy Awards. This information can be found in the file *"best_picture.txt"*. Your assignment is to use regular expressions to find important information such as movie titles, phrases with certain keywords, dates, and URLs.

To do so, you will complete the following functions in HW5.py:

1. find_movie_titles(string_list)

This function returns a dictionary with the keys being numbers (1 - 8) and the values being the names of movies. This function should use a regular expression to find the movie title pattern and then add each movie title to a dictionary with the keys representing that movie's position in the list of nominees and the values being the movie titles themselves.

The expected output should be in the format:

{1: "Belfast", 2: "CODA", … }

2. find_and_phrases(string_list)

This function finds all phrases with the word *"and"* (for the purposes of this assignment, a "phrase" includes the word *"and"* and the words immediately before and after it) and then returns them in a list.

For example, in the following text:

title: King Richard
https://www.kingrichardfilm.com/
Director: Reinaldo Marcus Green
Release date: 11 19 21
A look at how tennis superstars Venus and Serena Williams became who they are after the coaching from their father Richard Williams.

The expected output should be in the format:

["Venus and Serena"]

3. find_urls (string_list)
This function finds and returns all the hidden urls which match the following conditions:

1. The URL should start with “http://” or “https://”
2. Followed by a “www.”
3. Followed by any characters except for whitespace
4. It should contain a “.com”

These are examples of valid URLs:
https://www.pythex.com
https://www.youtube.com/watch?v=dQw4w9WgXcQ

4. find_valid_release_dates (string_list)
This function finds and returns dates from a text file that match a regular expression. You will write the regular expression. A valid date is any date that follows any of the following formats:

mm/dd/yyyy
mm/dd/yy
mm-dd-yyyy
mm-dd-yy

Any date that does not follow any of the above formats should not be returned from this function. For example, 121619 is not a valid date and should not be returned. Also, be careful that 08/32/2021 is not a valid date! Day should range from 01-31, and month should range from 01-12. Years should range from 1900 - 2021.

5. Make at least 3 test cases for **find_movie_titles**, **find_and_phrases, find_urls**, and **find_valid_release_dates**.

Count table

To help you know whether your functions are correctly implemented or not, here are the number of items that SHOULD be returned when you run your functions on the file *"best_picture.txt"*. To run your functions, you will first have to use the provided function ***read_file*** to convert the contents of *"best_picture.txt"* into a list of strings.

Note: your functions should return a list/dictionary, not the number of items (except for the extra credit function). This table is for you to verify that you are returning the right list (e.g., maybe you can check the length of the list you return):

Data type	**Number of appearances in the text**
Movie Titles	8
Phrases with "and"	10
URLs	7
Release Dates	5
Extra credit: Count "is" Count "a"	2 89

Grading Rubric (60 points)

This rubric does not show all the ways you can lose points.

- 6 points for creating tests for **find_movie_titles** (2 points per test case, up to a max of 6 points)
- 9 points for correctly implementing **find_movie_titles**
- 6 points for creating tests for **find_and_phrases** (2 points per test case, up to a max of 6 points)
- 9 points for correctly implementing **find_and_phrases**
- 6 points for creating tests for **find_urls** (2 points per test case, up to a max of 6 points)
- 9 points for correctly implementing **find_urls**
- 6 points for creating tests for **find_valid_release_dates** (2 points per test case, up to a max of 6 points)
- 9 points for correctly implementing **find_valid_release_dates**

Extra Credit (3 points):

Write a function ***count_mid_str(string_list, string)*** to return a count of the number of times a specified string appears in a file. It should match the string that is in the middle of a word (not the beginning nor the end). For example, if called with “be” it should match “number” but not “vibe”. Make sure to account for punctuation (e.g ‘,’ ‘?’) in your regular expression. You MUST use a regular expression to earn credit for this part. (We will not be checking if you make tests for the extra credit, but feel free to write your own tests if it will help you complete this problem!)

Submission:
Make at least 3 git commits and turn in your GitHub repo URL on Canvas by the due date to receive credit.