Assignment #1—Shell We?

Due: January 23, 2023 at 11:59pm

This assignment consists of two different components:

1. **You'll familiarize yourself with RegEx by completing a few exercises**
2. **You'll combine your RegEx and shell expertise to wrangle some words**

We expect this assignment to take 1-3 hours depending on your proficiency level with the tools. If you find yourself unproductively stuck or unproductively struggling (spinning in circles), ask on Ed and/or go to office hours!

As a reminder, you can use `man` pages to learn more about commands you don't yet know (especially what flags they support). The website cheat.sh also has many examples of how to use different commands.

Learning Goals
We think it's important that the assignments that we give exercise the tools we're learning in class–by completing this assignment, you'll have grown one assignment's worth of proficiency with each of these tools and systems (yay!!). Here's what you can expect to take away from this assignment:

- You'll have grown significantly more experience using the shell: in particular,
    - you'll get lots of practice not only with the utilities we've talked about, but also
    - with chaining them together and pushing their output around.
- You'll have grown in your ability to discover information about the various tools and utilities available to you. This is super important, because that's one of several ways you'll be self-teaching yourself these tools "in real life."
- You'll be pretty familiar with parsing real data using RegEx and the various shell tools.

Part I: Write Simple Regular Expressions (1.5 points)

For this part of the assignment, you will get some practice writing simple regular expressions. You should complete up through Lesson 10 in the regular expressions tutorial found here.

Place all of your answers in a simple text file called `regex.txt`. Remember that you can make a new text file using the `touch` command. You can open a file by using the `open` command (on macOS), the `explorer.exe` command (on Windows) or the `xdg-open` command (on Linux). The answer for each lesson (including lesson 1½) should go on a new line inside of `regex.txt`. *Note that there are many possible correct answers to each exercise!*

Important: Make sure you name your file `regex.txt` and not something else.

Part II: Write Data Wrangling Pipelines (1.5 points)

For this part of the assignment, you will get some practice with `sed` and `grep` by writing commands to parse data inside a dictionary file we provide, which is a newline-delimited list of English words.

Each exercise in this part can be solved multiple different ways (as is often the case with data wrangling!).

To begin, use the command below to download a dictionary file. (Different machines have different dictionaries installed, so we want to make sure everyone is using the same one):

```
Unset
curl -Lo dictionary.txt
https://cs45.stanford.edu/res/assign1/dictionary.txt
```

You can run `cat dictionary.txt` to see what the dictionary of words looks like.

First, find the number of words in `dictionary.txt` that have at least three `o`'s. Take a look at the `man` page of `grep` or `sed` for possible ways to do this.

Next, find the number of words in `dictionary.txt` that have **exactly** three `o`'s *and* where all `o`'s are separated by at least one non-`o` character. In other words, you should match a word such as `microbiology` (as all three `o`'s have at least one non-`o` character) but you should not match a word such as `zootrophy` (as there are two adjacent `o`'s that are not separated by a non-`o` character). This exercise is a little tricky as you need to consider that some words begin with an uppercase `O` and regexes are case sensitive.

Finally, find the three most common last three letters for words in `dictionary.txt` that have two adjacent `o`'s. In other words, find a list of all words in `dictionary.txt` that have two adjacent `o`'s. (The words can have more than two `o`'s as long as at least two of them are adjacent). Then find the three most common final three letters of these words. (Hint: take a look at how you might use `grep` to strip off the final three letters and how you might use `sort` and `uniq` to find the most frequent occurrences for a given set of data).

To submit your answers, create a file named `words_commands.txt` and place the commands you used to generate your answers for each subpart on a new line.

Feedback Survey (0.5 points)

Once you have completed the assignment, you can earn an additional 0.5 points by completing our anonymous feedback survey. Given this is the first offering of the course, we want to collect as much feedback as possible to improve the course in the future. You can complete the survey here. Once you complete the survey, you will receive a completion code which you should place in a text file named `survey.txt`. The survey is anonymous so submitting the completion code is the only way to verify that you completed the survey. *Please do not share this code with anyone, as that would constitute a breach of the honor code.*

Submitting Your Assignment

Once you have finished this assignment, you will need to upload your files to Gradescope. Make sure to upload both files to the Assignment 1 submission page. You should also upload `survey.txt` if you completed the survey.

All files must have the same name as specified above.