CSE 391, Spring 2019
Homework 7: Regular Expressions
Due Tuesday, May 28, 2019, 11:59 PM

This assignment focuses on using regular expressions and related commands such as `sed` and `egrep`. A set of files you will need for this assignment are available in the file `hw7.tar.gz`, found on the Homework page.

Submit your answers to the Google Form. Note that you can return to the Google Form and modify and re-submit your answers multiple times before the deadline. In response to each question, **write the command that will perform the task described**, *not the output that the command produces*. Please be sure to write the **entire command** (including the command name and the input file).

Task 1: Bash Shell Commands with `egrep`:

For each item below, **determine a single `bash` shell statement that will perform the operation(s) requested**. Each solution must be a one-line shell statement, but you may use input/output redirection operators such as `>`, `<`, and `|`. If you get the listed # of words, count yourself done. Note that the smaller words file found in `hw7.tar.gz` only has words up to 8 characters long so it will not work for all of the questions; using `/usr/share/dict/words` is recommended there.

Note: If you are running on an ***older*** version of the CSE VM, you may need to install the `/usr/share/dict/words` file you will use for this homework by typing this at a terminal command line **(the 19wi version of the VM and `attu` already have this file installed)**:

```
sudo yum -y install words
```

For these problems, write a command that uses `grep -E` or `egrep` to list the given words from the file `/usr/share/dict/words`, one word per line. Use regular expressions as appropriate. Use a single `grep` call per command.

1. All words that contain a 'z', followed by any single character, followed by an 'x'.
(On the CSE VM/`attu` there are 22 of them, from "azox" to "zaxes").

2. All words that contain the text "banana" or "mango".
(On the CSE VM/`attu` there are 21 of them, from "banana" to "mangour").

3. All words that contain a string of at least 5 lowercase vowels (a, e, i, o, or u) **in a row**.
(On the CSE VM/`attu` there are 9 of them, from "cadiueio" to "queueing").

4. All words that end with "never". (The word must *end* with "never", not just contain the substring "never".)
(On the CSE VM/`attu` there are 4 of them, from "minever" to "whenever").

5. All words that are exactly 25 **letters** long, in **reverse ABC order**.
(On the CSE VM/`attu` there are 8 of them, from "superincomprehensibleness" to "antidisestablishmentarian").

6. All words that start with either 'q' or 's', and that also contain a double z ('zz') later in the word.
(On the CSE VM/`attu` there are 66 of them, from "quizzability" to "swizzling", note that words may contain letters or -).

7. All 5-**letter** words that start/end with the same 2-letter substring.
(This one can take a long time to run on the `words` file on the CSE VM/`attu`, so you may want to use the shorter `words` file included in `hw7.tar.gz`. In that file there are 8 of them, from "edged" to "verve". On the CSE VM/`attu` there are 51 of them, from "abwab" to "zanza" if you only allow **letters** in words). Hint: back-references will be useful here.

Hints: Recall the **regular expression syntax** shown in class, such as a `.` for any character, `|` for "or," `[]` for character classes, `{}`, `*`, and `+` for quantities, `^` and `$` for specifying the start/end of a line, and `\<` and `\>` for specifying the start/end of a word. Also note that `grep` can use "back-references" to refer to sub-patterns captured previously between `(` and `)`, such as `\1` for the first captured sub-pattern, `\2` for the second, etc.

Task 2: Bash Shell Commands with `sed`:

For each of the next few problems, write a command that uses `sed` (preferably with the `-r` command-line argument to enable full regular expressions) to search and replace text based on regular expressions. For some problems, you may need to combine `grep`/`egrep` and `sed` using `|` . Each command should use **at most one call** to `sed`, but you may use input/output redirection operators such as `>`, `<`, and `|` to combine it with other commands as needed.

Your commands **should not create any temporary files** during their execution. Feel free to somewhat match your answers to the actual file you are given (e.g. only one word per line); your regexes do not have to work for a more general case).

8. Output the contents of the file `email.txt` with all spaces replaced by dashes (-).

9. Output the contents of the file `Questions.java` with all occurrences of the word "public" replaced with the word "private". (Don't match words that have "public" as a substring, such as "publicly" or "republic".)

10. Suppose that a writer named Violet uses too many exclamation points. Output the contents of the file `v.txt` but with all occurrences of one or more `!` marks in a row replaced with two question marks. (For example, `!` would become `??` and `!!!!!!` would all be replaced by just two question marks `??` .)

11. Java programs can contain single-line `//` comments and multi-line `/*` ... `*/` comments. Sometimes a programmer uses a multi-line comment syntax, but the comment only occupies a single line.
Write a command that finds `/*` ... `*/` comments in `Questions.java` **that occupy a single line** and replaces them with a `//` comment. For example, `/* hello there */` would become `// hello there` . (Your command doesn't need to modify comments where the `/*` isn't on the same line as the `*/` . You may assume that any given line contains at most one comment.)

12. Europeans format their dates differently than Americans. Where we would write a date such as "May 12, 2010", they would write it as "12 May 2010". Output the contents of file `dates.txt` but with all dates changed from USA format to European format. Don’t worry about coming up with a fancy regex to match only legal months or only legal days of the month.

13. Convert all 10-digit phone numbers in the file `phone.txt` to 5-digit internal extension numbers.
(For example, `Abba, Cadabra     x67890` and `Timss, Aaron     x62859` .)

14. Give a modified version of #13 that takes the file `phone.txt` as input and displays the phone extension first, then 3 spaces, then the person's name.
(For example, `x67890   Abba, Cadabra` and `x62859   Timss, Aaron` .)

15. All 12-**character** words in the `words` file that start with 'p' through 's' and end with "ker", in *Pig Latin*.
In other words, show all of the characters of the word other than the first character, followed by a dash `-`, followed by the first character followed by "ay". Note: words can consist of letters or a -.
(On the CSE VM/`attu`, there are 25 of them, from "atternmaker-pay" to "witchbacker-say".)
(*Hint:* Use `grep` or `egrep` first to reduce the file to the words of interest, before running `sed` on those words.)

Hints: The regular expression syntax hints from the previous section on `grep` also apply to `sed`. Also recall that some special characters must be escaped by a \ backslash to be used in a regex pattern. Remember to put a 'g' at the end of your pattern, such as `s/oldpattern/newtext/g`, to process all matches on a line.