Assignment 1: Bash Shell Scripting & Regular Expressions

Due. Saturday, June 15th, 2019

In this assignment, you will practice shell scripting and writing regular expressions. You may work on this assignment with your peers (e.g., pair-programming), however, EACH student must turn in their own assignment.

What To Do

Part 1: Bash Shell Scripts

You will write two scripts to manipulate data in a text file. Both of your scripts will produce the same output, however, one script will use basic bash shell commands and the other will use the UNIX command 'awk'.

Step 1. Download the data file 'studentGrades.txt' from Canvas.

Step 2. Create a bash script file with the name 'grades.sh' and another bash script file with the name 'gradesAwk.sh' that do the following:

a. Reads the contents of 'studentGrades.txt'. You should read in the name of the data file from the command-line arguments. The file names should not be hard coded in the scripts! We will test the scripts with additional data files that have different records.
b. Calculates the average of the scores for each student record. Rounded or truncated averages are accepted. Other data files that we will test your scripts with will also have exactly 3 grades per record.
c. Sorts the output by average score in reverse order (i.e., high to low or descending order), then by last name (i.e., ascending or default order), and then the first name (i.e., ascending or default order), then by ID (i.e., ascending or default order).
d. Formats the output as shown below:

```
99 [290010111] Lee, Terry
93 [928441032] Forester, Chris
92 [888111818] Forney, JC
91 [999999999] Smith, Jaime
82 [928441032] Forester, Jess
82 [123456789] Johnson, Lee
81 [199144454] Camp, Tracey
```

80 [434401929] Camp, Skyler
71 [299226663] Camp, Laney

****Note: The ID field is unique, however, student records may have the same scores, last name, and first name. Your 'grades.sh' script may NOT use the UNIX 'awk' command. Your 'gradesAwk.sh' script MUST use the UNIX 'awk' command.**

Step 3. When your scripts are run without the text filename 'studentGrade.txt' as the command-line argument, print out a usage statement. E.g.,

Usage: grades.sh filename
OR
Usage: gradesAwk.sh filename

Step 4. At the top of your scripts, include a comment with your name (and your partner's name if you pair-programmed with someone else).

Part 2: Regular Expressions

You will create a shell script with regular expressions that answer some questions about a data file.

Step 1. Download the data file 'regexData.txt' from Canvas.

Step 2. Create a file named 'regexAnswers.sh'. Write the regular expressions necessary to answer the questions below in 'regexAnswers.sh'. Write one regular expression per line. If you do not know the answer to a question, use 'echo "0"' so that your answers align with the questions (i.e. the regular expression for question 4 should be the 4th regular expression in the file).

Questions

1. How many lines end with a number or an alphabetic letter?
2. How many lines do not start with a vowel?
3. How many lines have 12 or more alphabetic letters?
4. How many lines contain only numbers? (e.g., 4, 32, 94932, etc.)
5. How many email addresses are from UCDenver? (e.g., end with 'UCDenver.edu')
6. How many phone numbers are in the data file (i.e., format: _ _ _ - _ _ _ - _ _ _ _) ?
7. How many city of Boulder phone numbers are in the data file? (e.g., starting with 303 - _ _ _ - _ _ _ _) ?
8. How many lines begin with a vowel and end with a number?

9. How many email addresses are in ‘first.last’ name format and involve someone whose first name starts with a letter in the second half of the alphabet? (e.g., first half: A - M or a - m, second half: O - Z or o - z)

Step 3. When your scripts are run without the text filename ‘studentGrade.txt’ as the command-line argument, print out a usage statement. E.g.,

Usage: regexAnswers.sh filename

Step 4. At the top of your scripts, include a comment with your name (and your partner’s name if you pair-programmed with someone else).

****Hints: The commands grep and egrep treat braces (i.e., { }) differently. Make sure you check for word boundaries (e.g., ‘\b’) where appropriate.**

What To Turn In

You will submit a single ZIP file containing all three scripts and upload it to Canvas. EACH student must submit their own ZIP file.