CSE 337: Homework Assignment 2

Instructions

Please read the the following instructions carefully before coding. You may lose points if you fail to follow these instructions.

- Deadline for this assignment is **October 28, 11:59PM. No late submissions accepted**.
- Stick with built-in Perl modules unless otherwise specified
- Keep your answers for each question in a separate file, and specify the file name clearly (*e.g.,* q1.pl for question 1). For the questions with multiple parts, you can use a filename like q1_p1.pl
- Make sure your programs work in other machines. One way to test this is to test your program in our UNIX servers, and see if it works there. You will lose marks if your program fails to run in our machines.
- Put all of your Perl files, and necessary input files in a single folder (Do not make a sub-folder for each question).
- **The questions will be graded using a script unless otherwise specified. Hence, please strictly adhere to the sample input-output formats provided.**

  Note: Please rename your folder in the right way before zipping! For example, if your name is Ying Lu and your student ID is 111345678, put all submission files in a folder named Ying_Lu_1112345678, then compress that folder to Ying_Lu_1112345678.zip.

1 Preparing an Exhibition [20 pts]

You get a chance to work with art museum staff. They have a file named 'collections.csv', which they use to keep a record of all their artworks. The curators are now preparing an exhibition and they need your help. Here is a sample from 'collections.csv':

```
ID,Name,Country,Year
10,Dish with Flowers and Birds,China,1561
22,Wall light (one of a pair),France,1662
35,Water cooler,America,1850
44,Vase with flowes, China, 1709
```

The first line of 'collections.csv' gives description of whole file: artwork ID, artwork name, the country artwork came from, and the year the artwork was created. The ID is unique for each artwork. It is ensured that all artworks are recorded in ID-increasing order, country name always begin with an uppercase letter followed by lowercase letters, and all years are valid numbers.

All csv files provided are just samples. Your program will be tested on completely different csv files with the same format.

Part 1. Write a program to help curators quickly find out how many artworks are there from a certain country.

Use standard input(keyboard input) to get the country name. Your program should print the number of artworks from that country to the screen. Country names are case sensitive. [6pts]

```
Sample input:
China
Sample output:
2
------------------------------------------
Explanation:
There are two artworks from China:
10,Dish with Flowers and Birds,China,1561
44,Vase with flowes, China, 1709
```

Part 2. The curator wants to create a timeline to help visitors better understand all artworks. Find the oldest and newest artwork and print out their ID on screen. The output should have two lines, each line has one single number. The first line should contains ID of oldest artwork, the second line should contains ID of newest artwork. It is ensured that there will only be one oldest artwork and one newest artwork.[6 pts]

```
Sample output:
10
35
-------------------------------------------
Explanation:
The oldest artwork is No.10 created in 1561
The newest artwork is No.35 created in 1850
```

Part 3. Exciting news: two other museums will collaborate with you to prepare an exhibition together. They have provided their records in 'm1.csv' and 'm2.csv'. All csv share same format. Help curators merge ‘collections.csv‘, ‘m1.csv‘ and ‘m2.csv‘ together into a new file ‘exhibition.csv‘. Painting IDs must be organized in increasing order. Input ensure all IDs are unique, so you don't need to deal with ID conflicts when merge three csv. [8 pts]

```
Input sample:(see ‘collections.csv‘, ‘m1.csv‘, ‘m2.csv‘)
Output sample:(see ‘exhibition.csv‘)
```

2 Text Transformation[15pts]

Suppose you are given a text file ‘q2.in‘, a piece of text (a sequence of lines). A sample text looks like this:

```
O Captain! my Captain! our fearful trip is done,
The ship has weather’d every rack, the prize we sought is won,
```

Assume that the words are separated by spaces.
Part 1. Write a subroutine to swap str1 with str2 in ‘q2.in‘. Your program should use standard input(keyboard input) and print results to a file named ‘q2p1.out‘. Input contains two lines, first line is str1, second line is str2. All strings and words are case senstive.

```
sample input:
Captain
shirley

sample ‘q2p1.out‘:
O shirley! my shirley! our fearful trip is done.
The ship has weather’d every rack, the prize we sought is won,
---------
Explanation:
str1 is "Captain", str2 is "shirley".
your code should be able to change all "Captain" into "shirley".
```

Part 2. Write a subroutine to remove lines that has exact $K (K \geq 1)$ words in ‘q2.in‘. If there are no such lines, print out "Oooh Nooo!". Your program should use standard input(keyboard input) and print results to a file named ‘q2p2.out‘. Input only has one line containing integer K.

```
sample input:
9
sample q2p2.out:
The ship has weather’d every rack, the prize we sought is won,
------
Explanation:
The first line has 9 words:
O shirley! my shirley! our fearful trip is done.

sample input:
2
sample q2p2.out:
Oooh Nooo!
------
Explanation:
Both lines in q2.in have more than 2 words.
```

3 Array Operators and Hashes [20 pts]

Part 1. Write a program that takes a text file as input(as a command line argument). This file contains a set of words in each line and a feature(an integer) associated with this set of words, separated by a comma. The feature varies from 0 to 9, inclusive.

Please print out the total number of words with a particular feature value. The output should be in the following format:

```
$ perl q3_p1.pl features.txt
total_#_of_words_with_0
total_#_of_words_with_1
total_#_of_words_with_2
..
..
total_#_of_words_with_9
```

Part 2.
The following table has the revenue(in dollars) of a small company for each month in the year 2017:

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Revenue ($)	4840	4340	3900	4330	3090	3660	3520	3280	4130	3690	4260	4800

You should write a program that calculates the cumulative revenue of two months given by the user. The three letter month abbreviations **are not** case sensitive. The program **must use a hash** of revenue and be able to handle invalid input, otherwise keep asking for the months. Please, read carefully the explanation of the sample inputs and outputs for further details. [10 pts]

```
$ perl q3_p2.pl
Enter the initial month: Feb
Enter the final month: may
The cumulative revenue is: 15660
------------
Explanation:
The cumulative revenue depends on "Feb", "Mar", "Apr" and "May".

$ perl q3_p2.pl
Months can be selected using the three initials letters.
Enter the initial month: apr
Enter the final month: July
Please enter only the three initials letters of a valid month.
Re-enter the final month: uly
Please enter only the three initials letters of a valid month.
Re-enter the final month: jUL
The cumulative revenue is: 14600
------------
Explanation:
The program keeps asking for the correct final month initials.
Then, it shows the cumulative revenue that depends on "Apr", "May", "Jun" and "July".

$ perl q3_p2.pl
Months can be selected using the three initials letters.
Enter the initial month: june
Please enter only the three initials letters of a valid month.
Re-enter the initial month: jun
Enter the final month: feb
Please enter only the three initials letters of a valid month.
Re-enter the final month: jul
The cumulative revenue is: 7180
------------
Explanation:
The program keeps asking for the correct initial month initials.
"Feb" can not be the final month given "Jun" as the initial month.
Then, it shows the cumulative revenue that depends on "Jun" and "Jul".
```

```
$ perl q3_p2.pl
Enter the initial month: Oct
Enter the final month: oct
The cumulative revenue is: 3690
------------
Explanation:
The cumulative revenue depends on a single month.
```

4 Files and Directories [20 pts]

In this question, you are asked to split the feature values from question 3 part1 into 10 different files as follows:

1. Create a directory named "features".
2. Inside "features" directory, create 10 files with the following format: `"N-features.txt"`, where "N" is an integer from 0 to 9.
3. From the file `"features.txt"`, read each line and add it (without the feature value) to the corresponding `"N-features.txt"` file. For example, the file "3-features.txt" will only contain the lines with a feature value of 3.
4. Once done, your program should print "Files have been created!" and in the next 10 lines print the full path of each file.

```
$ perl q4.pl features.txt
Files have been created!
"print full path of 0-features.txt here"
"print full path of 1-features.txt here"
"print full path of 2-features.txt here"
..
..
..
"print full path of 9-features.txt here"
```

Please do not create any file or folder manually and do not attach any of the created files in this question. Your program will be tested on a different dataset.

5 Regular Expressions [25 pts]

In this question, each subsection contains different types of regular expression questions.

5.1 Matching strings[4 pts each]

For each item below, write a regular expression that matches the strings on the left, while not matching the strings on the right. **Note:** The regular expression should match entire strings, not just the beginning/end of it.

be or not to be	to be or
to be to be	to be
to be to be to be	to be or not
to be to be to be to be	to be or not to

hello	cat.
world!!	dog?
hello!?	computer..
perl?!	world!

For submission, create a text file named "q5_p1.txt", and provide your regular expression solution for each item in a new line(*i.e.,* 2 lines in total).

5.2 More Regular Expression Questions[4 pts each]

- Write a regular expression that matches the following format: Month/day/year(For example: "1/25/2018", "3/11/2119", "6/8/224", but not "6/54/1996"). The regular expression you wrote should capture month, day and year information as separate groups. To keep things simple, you can assume there are 30 days in every month, and the year field has at most 4 digits.
- Write a regular expression that matches a sentence in which every word has less than 5 characters. Note: The last word cannot be followed by a space.

For submission, create a text file named "q5_p2.txt", and provide your regular expression solution for each section in a new line.

5.3 Finding examples and counter-examples[3 pts each]

For the regular expressions presented below, provide **three** strings that matches the regular expression, and **three** strings that do not match the regular expression.

- `[-+]?\d*(\.\D+)?F\s`
- `(#?)(1?)(one)\1\2\3`
- `((a*?)\b).*\w\2\1`

For submission, create a text file named "q5_p3.txt", and provide your examples by indicating whether it matches or not. This part will not be graded by a script, so you are allowed to use whichever format you want, as long as you make your answers clear.