o grep -c –E '(apple|orange)' c.txt
o uniq -c c.txt | sort
o sort c.txt | uniq -c
o sort -u c.txt | uniq -c

Please complete Question 1 on Moodle. You do not need to submit any file for this question.

Question 2 [80%] Shell programming

7-Eleven is the largest convenience store group in Hong Kong, which offers a wide variety of products and convenient services. Sales records in each convenience store are collected from a central backend system for analysis. The system generates log files every day and each line in the log file represents a transaction made through a specific convenience store and has the following format:

```
[Timestamp], [ProductID]
```

• Timestamp has the format YYYY-MM-DD HH:MM:SS.Millisecond.
• ProductID is an 8-digit value identifying a product provided by the company.
• Comma "," is used as field separator in the log file.
• The log files of 7-Eleven company have "7Eleven_" as prefix in their filenames, followed by the store ID of 5 digits identifying a store location, followed by the date in YYYY-MM-DD, followed by ".log".

Consider an example 7Eleven_00123_2016-01-01.log file that records the transactions of the store with ID "00123" on 2016-01-01:

```
2016-01-01 15:44:41.134203,57553901
2016-01-01 15:50:36.294620,86782909
2016-01-01 15:50:36.294620,86782909
2016-01-01 15:50:36.294620,86782909
2016-01-01 15:50:36.294620,86782909
2016-01-01 15:50:36.294620,86782909
2016-01-01 16:02:14.628319,15527051
...
2016-01-01 17:14:29.360980,86782909
2016-01-01 17:17:27.849505,15461586
...
```

• You can assume that the transactions are sorted in ascending order of the Timestamps.
• From the log file we can trace a product's transactions in a day. E.g., the six transactions of product 86782909 are highlighted as a running example. The first five transactions have the same timestamp, which means the customer bought these 5 items at the same time.

1. Create a shell script **analyze.sh** that do the following:

• For each "7Eleven_00123_" log file, output item counts and the top three product IDs with the most amount of items sold in descending order of item counts.

• You can assume that all the "7Eleven_00123_" log files are stored in the same directory of **analyze.sh**.

• If the number of items were the same for two different product IDs, we output them in descending order of the product IDs. Therefore "9 81060085" is put ahead of "9 78588321" under "7Eleven_00123_2016-01-01.log" in the example below.

• If we run **analyze.sh** on the directory that consists of log files from 2016-01-01 to 2016-01-05 we provided, the following will output:

```
7Eleven_00123_2016-01-01.log:
        10 28468614
        9 81060085
        9 78588321
7Eleven_00123_2016-01-02.log:
        10 91743551
        10 55056411
        10 37290775
7Eleven_00123_2016-01-03.log:
        10 54293095
        10 45573708
        9 41965222
7Eleven_00123_2016-01-04.log:
        10 28445733
        9 82882899
        9 78096304
7Eleven_00123_2016-01-05.log:
        10 22184530
        9 93683090
        9 91743551
```

Hints:

- You can make use of the commands "sort" and "uniq" to get the number of occurrences of each product.
- For the "uniq" command, using the flag -c can return the row count.
- You can make use of the command "head -n x" to return the first x row.
- After you have finished the shell script, you are strongly suggested to create some other input data to test if you analyze.sh is correct. E.g., Updating 7Eleven_00123_2016-01-01.log to make 10 transactions for product 86782909, to see if you are able to output the correct result.

2. Suppose that besides the "7Eleven_00123_" log files, we also get the sales record of the store with ID "00456". The log files of the store 00456 have the same format as those of store 00123. For instance, consider the 7Eleven_00456_2016-01-01.log file that consists of all the sales records made in the store 00456 on 2016-01-01:

```
...
2016-01-01 14:54:27.733836,57553901
2016-01-01 14:8:46.747141,86782909
2016-01-01 14:8:46.747141,86782909
2016-01-01 14:8:46.747141,86782909
2016-01-01 14:8:46.747141,86782909
2016-01-01 14:8:46.747141,86782909
2016-01-01 14:8:46.747141,86782909
2016-01-01 15:1:7.106278,90056244 ...
```

• You can assume that the transactions are sorted in ascending order of the Timestamps.

• You can also assume that the product ID is unique for each product across the log files of all 7-Eleven stores.

• From the log files we can trace a product's transactions in various stores. E.g., the twelve records of product 86782909 (Six in 7Eleven_00123_2016-01-01.log, and six in 7Eleven_00456_2016-01-01.log) are highlighted as a running example.

Create a shell script **trace.sh** that does the following:

• **trace.sh** takes one command line input argument. The input argument represents the ProductID that we would like to trace.

• You can assume that the "7Eleven_00123_" and "7Eleven_00456_" log files are all located in the same directory as trace.sh. If we run **trace.sh** as follows:

```
$ ./trace.sh 86782909
```

trace.sh generates a file 86782909.log that contains a list of sales records with productID 86782909 in all "7Eleven_00123_" and "7Eleven_00456_" log files, sorted in ascending order of the date and timestamp.

• As an example, 86782909.log consists of the following transactions:

```
2016-01-01 14:8:46.747141,86782909
2016-01-01 14:8:46.747141,86782909
2016-01-01 14:8:46.747141,86782909
2016-01-01 15:50:36.294620,86782909
2016-01-01 15:50:36.294620,86782909
2016-01-01 15:50:36.294620,86782909
2016-01-01 15:50:36.294620,86782909
```

We only show the seven transactions of product 86782909 in our running example.

• Note that **trace.sh** will only create the trace log but output nothing except the following cases:

o If the number of input arguments is not 1, then **trace.sh** outputs "Usage: ./trace.sh (ProductID)" in shell prompt.

o If there are no sales records found, then 86782909.log will be deleted. trace.sh will output "No records found for 86782909" in shell prompt.

- Please run **trace.sh** multiple times and check the correctness of the created log file.

–END–