Big Data Analytics (CSCI-720)
Programming Project

This assignment is due by 3:00 pm on Thursday December 3 to the myCourses dropbox.

You may work with one other student in this class on this assignment (team of two people). Otherwise this assignment is to be completed as an *individual effort*. **Code submissions may be subjected to an originality check.**

Your solution MUST run on the CS department machine glados.cs.rit.edu when invoked using the command `python3`. Solutions that do not run on glados will not be given any credit. You can remotely access glados using ssh (or other similar tools).

Go back and read the previous paragraph again. If your solution does not run under python3 on glados, you will not be given any credit for the assignment.

Overview

You are provided a file of transaction data from a local grocery store. Sample data:

[8405 , yogurt bread chicken flour grapefruit oliveOil cottageCheese]
[9897 , onions]
[7417 , tomatoes onions frozenWaffles rice sugar]

Here, customer # 8405 purchased 7 items; customer # 9897 purchased only onions, and customer # 7417 purchased 5 items. Notice that 2-word items are handled as a single token – for example cottageCheese or frozenWaffles.

You may also see the transaction shown below indicating that the customer voided the transaction:

[7847 ,]

The store wants to know which items pairs are typically purchased in the same transaction. Looking again at this transaction:

[7417 , tomatoes onions frozenWaffles rice sugar]

These are the item pairs in this transaction:

"tomatoes/onions"
"onions/frozenWaffles"
"tomatoes/frozenWaffles"
"rice/sugar"
"rice/tomatoes"
"tomatoes/sugar"
"onions/rice"
"sugar/onions"

"frozenWaffles/sugar"
"rice/frozenWaffles"

Notice that "sugar/onions" and "onions/sugar" represent the same item pair and should not be reported twice.

The marketing department wants to use this information to determine potential products to display on the store's endcaps. You have all seen endcaps – the image below shows an endcap where a retailer is displaying crackers with peanut butter (Image source: discountshelving.com/images/storetype/grocery/Grocery-Store-Gondola-Endcap.jpg).

If you look closely in Wegmans (I assume everyone is familiar with Wegmans!) you may see endcaps displaying peanut butter with jelly, spaghetti with sauce, or toothbrushes with toothpaste – all items typically purchased together.

The use of endcaps is a common marketing technique; since there are relatively few endcaps in a store, retailers are always looking for the "biggest bang" for the space so they want to display items customers typically purchase together.

The marketing department has determined that product pairs must be purchased with a relatively high frequency in order to appear on an endcap – in this case, in at least 650 different transactions. Using the data provided, determine all item pairs occurring more than 650 times in the data file provided.

Note the following:

- It does not matter who (which customer number) purchased the item pair. For this problem, the customer number should be ignored.
- Item pairs must occur in the same transaction.
- Remember that "sugar/onions" represents the same item pair as "onions/sugar" so you should not report both.

Your item pairs occurring more than 650 times must be the only output displayed by your program. Be sure you remove all debug statements before you submit. In your output display only the item pairs meeting the criteria. Do not display the counts.

Sample output:

```
sugar/onions
onions/tomatoes
onions/frozenWaffles
```

… and so on

What to Submit

Submit your carefully written and commented code along with a brief writeup.

Code:

You will submit your code via 'try' – follow these instructions:

1. **Be sure your name appears in the file documentation header in your code.** Name your code ItemPairs.* I have shown a * for the file extension because you are free to implement your solution using any language. I have 'try' set up to accept Python (.py) and Java (.java) solutions. If you plan to use a different language, you must contact me ASAP so I can modify the 'try' control files or your submission will not be accepted! Please, if you are using a language other than Python or Java, do not wait until the last minute to inform me or I may not have adequate time to modify the control files prior to the due date and your submission will be rejected.

If you have not used 'try' before, be advised that it checks the submission time and will not allow a submission after 11:59 pm on the due date. When you submit, the last operation 'try' performs is to check the system time. If you start submitting at 11:58 and 'try' does not finish its internal operations until midnight, it will reject your submission

as late! The bottom line: Don't wait until the last minute to submit! If you submit multiple times, 'try' will archive only your most recent submission. My advice is to "submit early and often" in case something prevents you from submitting late on the due date. Note also that other classes may have projects due around the same time so the systems may be running slow. It is your responsibility to be sure your submission is received on-time so **plan ahead** and don't wait until the last minute! Late work will not be accepted. You must submit via 'try' – no emailed solutions will be graded.

2. Log onto a CS machine (such as glados).

3. Register in my grader account by issuing the following command:

```
try tmh-grd register /dev/null
```

This command is `try <SPACE> tmh-grd <SPACE> register <SPACE>  /dev/null`

You will be prompted for a lecture and lab section. Enter your lecture section number and 99 for the lab section. Be sure you enter your correct lecture section or your submission will be sent to the wrong directory and won't be graded. Be sure you enter either 02 or 03 for your lecture section! If you aren't sure of your lecture section, check your enrollment record in SIS.

4. Now submit your code using the command below to submit your carefully tested code solution:

```
try tmh-grd project1-EC ItemPairs.EXT    *see note below
```

*EXT = `ItemPairs.java (java solution) or ItemPairs.py (Python solution).`

I expect professional-level code so be sure you follow a coding standard, document your code, provide file header (with your name) and method/function documentation headers, etc.

If you develop your code on a non-CS department machine, be sure you allow time to transfer your files to glados and retest your solution. I will test your submissions on glados so your code must correctly execute that machine. No credit for this project if your code solution does not run on glados.

If you use Python with my MapReduce simulator, you do not need to submit MapReduce.py. This implies that if you use MapReduce, you do not modify the basic MR engine!

Writeup:

Submit your writeup file to the myCourses dropbox. Your writeup must be either a Word or .pdf file containing the 4 items below. Please be sure to spell and grammar check prior to submitting. DO NOT SUBMIT ANY OTHER FILE FORMATS!

Please, be sure your writeup is direct and to the point. Do not include extraneous screenshots or the like. Respond only to the questions listed below. Writing counts so be sure you carefully proof your work before you submit.

Include each of the following in your one page writeup. Please number each item in your report.

1. Your name OR your name plus your partner's name if you did not work alone.
2. 2-3 sentences explaining how you selected your target language.
3. 2-3 sentences describing how you feel your solution would execute if the amount of data increased 1000-fold in size. Would your implementation (considering data structures you may have used) still be usable?
4. 2-3 sentences describing any problems you encountered, or any known limits to your program.

Note that try will not attempt to run your code. It will check for the correct files and reject your submission if the file is not correctly named.

Grading

Your functional code will be worth 70% of your grade. If it does not compile and run on glados, you will not receive any credit. I will run your program and compare your output to the known solution.

The remaining 30% will be awarded based on your project report. Remember this is a significant portion of your course grade so be thorough with your explanations. Writing quality counts!

Rev. 10/21/2015