CPT 237 JAVA PROGRAMMING

Assignment 6 – Acme Cab Co. – North Charleston

## The problem

Well, as you may have expected, Donna’s nephew Leo hasn’t worked out so well. He has failed completely in his attempt to write a class to read the file. So, that is now up to you. In addition, Donna realizes that just looking at the screen isn’t enough to keep track of everything; she wants to see it in a spreadsheet. For our purposes, we will save the information to a .csv, a text file with all the values separated by commas. Donna would like to see most of the information that we saw on a per cab basis last week, with the exception of the longest distance between services. She wants the file to be a .csv with appropriate headers. It should match the following format EXACTLY.

Cab ID,Start Date,End Date,Gas Cost,Service Cost,Gross Earnings,Net Earnings,Miles Driven,Services Performed,Average Service,Max Service Days  
CAB24,9/2/2014,9/30/2014,105.23,50.00,300.50,212,145.27,2,70.67,12

The values are per cab, so there should be one line in the file for each cab.

There is no need to see the summary information. Donna will use your original program if she wants to see a summary…or simply let Excel do the work.

## Implementation notes:

There are two new parts to this program. The first is simply writing the code to read the cab record file. The classes that you expose should have exactly the characteristics described in last week’s assignment. That is, there should be a **CabRecordReaper** that has the **hasMoreRecords** and **getNextRecord** methods. The object returned by the **getNextRecord** should implement the **CabRecord** interface described last week. You should pass a string to the reaper’s constructor that indicates where to find the file. Note that you will not need to produce ‘fake’ records the way the demo jar file did.

The second part is simply to open a file for output and write to it. You will, of course, need to call your cab record to retrieve the values, put in the comma separators, etc.

To facilitate testing of this program, the input and output names should be in two named constants:

final String CAB\_RECORD\_FILE = “C:/path/to/input.csv”;  
final String CAB\_SUMMARY\_FILE = “C:/path/to/output.csv”;

You should be prepared to handle two types of errors (hint: Exceptions). First, the file from which you expect to retrieve the records may not exist. Your program should catch that, send an error to the console and exit. Second, you may get an error in your data file; either the date may not represent a real date or the numbers in the records may not be a real number. If you encounter a bad record, simply send a message to System.err, skip that record, and continue. Notice that these errors do not occur in the same place. An invalid number is caught in the reaper while an invalid date is caught further up in your program.

Note that the output file should match line for line the above format. I would like to be able to do a ‘diff’ between a correct file and your output and see absolutely no differences.

## Extra credit:

Log errors to a file. Include as much about the error as you can. E.g. the entire line in the case of bad numeric values, perhaps less in the case of a bad date. At the very least, the bad value as well as the cab id should be included.