IMPORTANT SUBMISSION INSTRUCTIONS

You will be uploading your submission using the assignment server. The link is posted to cuLearn. A short video is posted there also in case you require instructions. You may submit as many times as you wish, but must wait at least 5 minutes between submissions (to protect the server). Your highest mark is kept. It would help us out if you try the upload early, even using just the unmodified assignment skeleton, that way potential problems can be found before the deadline. It will also ensure that YOU know how to submit your assignment well before the deadline. If your attempt to obtain your secret code is unsuccessful, please email me at darrylhill@cunet.carleton.ca. This may happen if you have registered late, etc.

You must adhere to the following rules *(as your submission will be subjected to automatic marking system):*

- ***Download*** *the compressed file* ***"comp2402a1.zip"*** *from cuLearn.*
- ***Retain the directory structure*** *(i.e., if you find a file in subfolder "comp2402a1", you must not remove it).*
- ***Retain the package directives*** *(i.e., if you see "package comp2402a1;" in a file, you must not remove it).*
- ***Do not rename any of the methods already present*** *in the files provided.*
- ***Do not change the visibility of any of the methods provided*** *(e.g., do not change private to public).*
- ***Do not change the main method of any class****; on occasion the main method provided will use command line arguments and/or open input and output files – you must not change this behaviour).*
- ***Upload*** *a compressed file* ***"comp2402a1.zip"*** *to the assignment server to submit your assignment as receive your mark immediately* **(highest mark of all submissions will be your assignment mark).**

Please also note that ***your code may be marked for efficiency as well as correctness*** *– this is accomplished by placing a hard limit on the amount of time your code wil be permitted for execution. If you select/apply your data structures correctly, your code will easily execute within the time limit, but* ***if your choice or usage of a data structure is incorrect****, your code may be* ***judged to be too slow*** *and* ***it may receive a grade of zero****.*

You are expected to ***demonstrate good programming practices at all times*** *(e.g., choosing appropriate variable names, provide comments in your code, etc.) and* ***your code may be penalized if it is poorly written****. The server won't judge this, but in case of discrepancies, this will be evaluated.*

Instructions

Start by downloading the comp2402a1 Zip file from cuLearn, which contains a skeleton of the code you need to write. This assignment is about using the Java Collections Framework to accomplish some basic text-processing tasks. These questions involve choosing the right abstraction (Collection, Set, List, Queue, Deque, SortedSet, Map, or SortedMap) to efficiently accomplish the task at hand. The best way to do these is to read the question and then think about what type of Collection is best to use to solve it. There are only a few lines of code you need to write to solve each of them. Unless specified otherwise, sorted order refers to the natural sorted order on Strings, as defined by String.compareTo(s). Part 0 in the assignment is an example specification and solution.

COMP2402B (Fall 2019) – "Abstract Data Types / Algorithms"

Specification for Assignment 1 of 4

Part I of I – Coding

1. [5 marks] Read the input one line at a time until you have read all n lines and imagine these lines are numbered 0,...,n-1. Next output lines floor(n/2),...,n-1 followed by lines 0,..,.floor(n/2)-1.

2. [5 marks] Read the input one line at a time and output the current line if and only if it is smaller than any other line read so far. (Here, smaller is with respect to the usual order on Strings, as defined by String.compareTo().)

3. [5 marks] Read the input one line at a time and output only the last 9999 lines in the order they appear. If there are fewer than 9999 lines, output them all. For full marks, your code should be fast and should never store more than 9999 lines.

4. [5 marks] Read the input one line at a time and output the current line if and only if it is a duplicate of some previous line.

5. [5 marks] Read the input one line at a time. When you are done, output all the lines in case-insensitive sorted order.

6. [5 marks] Read the input one line at a time and output the current line if and only if it is not a suffix of some previous line. For example, if the some line is "0xdeadbeef" and some subsequent line is "beef", then the subsequent line should not be output.

7. [5 marks] Read the input one line at a time and output the current line if and only if you have already read at least 1000 lines greater than the current line and at least 1000 lines less than the current line. (Again, greater than and less than are with respect to the ordering defined by String.compareTo().)