CPSC2108 Data Structures, Spring 2014

ASSIGNMENT 3 – DUE BEFORE MIDNIGHT, WEDNESDAY, MARCH 12

The purpose of this assignment is to practice the application of the List Abstract Data Type (ADT) for problem solving.

The number of items in a list may grow so large that finding a value in the list not efficient. One way to improve efficiency is to maintain a number of smaller linked lists each of which can be directly accessed by using an index. The indexes themselves are stored in a direct access data structure such as an array.

Write a program to read lines of uppercase text from a file and to produce a simplified text concordance, which is a list of distinct words in the text, with their frequency (number of occurrence). Store distinct words beginning with A in one linked list, those beginning with B in another, and so on. After all the text lines have been read, sort each list and then print a list of all these words in alphabetical order with the frequency of each word next to in on one line. You can use any Java built-in dynamic (capable of growing) list class or create your own linked list to implement your solution to this problem.

What to submit:

- A single zip file containing the following:
  - ✓ The .jar file (make sure to include source code)
  - ✓ A Word or .rtf document showing screen captures of test runs.
  - ✓ Data file containing text you used for testing your program

Include all required items in a single zipped folder named *<your last name>_<your initial>_Assignment03.zip*. It is your responsibility to make sure you jar file can be extracted and the code runs on a different computer.

Submit your zip file using CougarView Assignment 3 drop box before midnight, Wednesday, March 12.

See next page for grading rubric.

Grading Rubric:

1	Good documentation (follow style given in textbook program listings) • Header documentation • Class and method documentation • Instructions to the user	10 points
2	Good programming style • Good use of white space • Good variable names • Good indentation	10 points
3	Robust • Will not crash with unexpected input • Will not crash if a data file is not found	10 points
4	Program • Meets all requirements • Good program logic • Good use of data structures • Efficient design	60 points
5	Sample runs • Screen shots of sample runs are included to adequately demonstrate the program	5 points
6	Instructions followed	5 points