Assignment 4
Sorting Problems

Release Time	Due Date
October 27, 2016	November 10, 2016

Objectives

- To practice bubble sort, binary-insertion sort, selection sort and merge sort.
- Practice developing high-performance solutions.
- Compare the differences of these 4 sorting methods.
- Start to get a (theoretical design) flavor of multicore shared memory solutions

Problem Specification

YG is a teaching assistant at WMU, and his supervisor asks him to sort students using alphabet (aka lexicographic or lexical) order in CS3310. Please help him develop a Java application to sorting the class list consisting of names / class-id [at most five digit] / primitive-Java-data-types. Since we may not know which sorting method is the best one, we have to try it. Here we go:

1) Read the data in *NameList.txt* file and store the names in a linked-list.
2) Sort the data in the linked-list with the 4 types of sorting methods (bubble sort, selection sort, merge sort, binary-insertion sort (i.e., insertion sort that uses binary search to find the insert position).
3) Besides string type, please let your application handle integer, float, double, char, in case we may need to use those in the future.
4) Please make your application could handle even billions of data (find out what is the maximum size your application can handle in case you can not get billion size to run at all on your machine).
5) Make sure your application is robust, readable, and has a friendly UI.
6) Analyze these 4 sorting methods (by time and space complexity) empirically as well as theoretically, and specify which is the best one to help YG when linked list is used to store data.

7) Repeat steps 1-6 using array-lists instead of linked-lists (find out what is the maximum size your application can handle in case you can not get billion size). For array-lists since you may not know the time for each insertion / deletion for array-lists, assume they are O(1) without loss in generality.

Design Requirements

Code Documentation

For this assignment, you must include documentation for your code as generated by JavaDoc. You should have JavaDoc comments for every class, constructor, and method. By default, JavaDoc should output html documentation to a subfolder within your project (/dist/javadoc). Make sure this folder is included when you zip your files for submission. You do not need to submit a hard copy of this documentation.

Hint: http://stackoverflow.com/questions/4468669/how-to-generate-javadoc-html-in-eclipse

Coding Standards

You must adhere to all conventions in the CS 3310 Java coding standard. This includes the use of white spaces for readability and the use of comments to explain the meaning of various methods and attributes. Be sure to follow the conventions for naming classes, variables, method parameters and methods.

Testing

Make sure you test your application with several different values, to make sure it works.

Assignment Submission

- Generate a .zip file that contains all your files, including:
    - Source code files
    - Including any input or output files
- Javadocs