Assignment 2

Sorting in arrays and linked lists

Release Date	Due Date
February 3, 2017	**February 19, 2017**

Objectives

- Experience various techniques to search and sort using arrays and linked-lists
- Practice developing high-performance solutions.
- Analyze the advantages and disadvantages of arrays and linked-lists
- Compare the space and time complexity between arrays and linked-lists.

Problem Specification

After first assignment, we already know how to efficiently search a value in a sorted array. This time, we will practice how to sort values in an array or a linked list (initially unordered).

Write a JAVA application to solve the following problem:

1) Generate n random floating-point numbers from 1 to m i.e., in the range [1, m)

2) Store the generated numbers in an array.

3) Sort the array using bubble-sort, merge-sort, quick-sort, insertion-sort, Java's built-in sorting to sort in a non-decreasing order, and output the result for each step (output only for n ≤ 100 and round-off the values to three digits after the decimal point.).

   (for example if the values were integers, then

   initial: 9,3,6,2,1

   after bubble-sort: 1,9,3,6,2

   after merge-sort: 1,2,9,3,6

   after quick-sort: 1,2,3,9,6

   after insertion-sort:1,2,3,6,9

   after Built-in-sort: 1, 2, 3, 6, 9)

   Note: Design your code so that it can also handle data types other than floating point numbers (e.g., integers, chars, Strings, booleans, etc.).

4) Analyze the time and space complexities for each algorithm (i.e. time complexity for bubble-sort is blah-blah, replace blah-blah with your answer). First give theoretical time complexities. Next plot the empirically observed time complexities, do a curve fitting, and compare with the theoretical. Do the two (theoretical vs empirical) agree? Why? Why not?

5) Redo step 2) to 4) with linked-lists.

6) Compare the advantages and disadvantages of arrays and linked-lists from your observations, then decide which data structure you will use for the following situations (just print your answer on the screen at the end of your program):

   a. Searching is the most frequently occurring operation.

   b. Insertions are the most frequently occurring operations.

   c. Store space is limited.

   (i.e., at the end of program: System.out.print("My answers are: a.?, b.?, c.?") replace ? with your answer)

7) Do not forget to repeat your experiments to get good empirical data

8) For 20 extra points:

   Implement one more sorting algorithm in addition to the above. Compare its time and space complexity against others.

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
- A single-pdf and source files of your Analysis Report.
- Don't forget to follow the naming convention specified for submitting assignments