Project 1 – Sorting

Overview

This project is to implement two sorting algorithms. The first sorting algorithm is an $O(N^2)$ algorithm: either Selection Sort or Insertion Sort. The second algorithm is an $O(nlogn)$ algorithm: either Mergesort or Quicksort. If you choose Quicksort you must use the median-of-three method to determine the pivot, and must use the Hoare Partition algorithm (both described in the book and slides). With all of this done, you will be asked to gather some data about the running time and plot it.

Requirements

You must implement the two sorting algorithms above. They should sort the numbers in ascending order (smallest to largest). When you submit your project, you must have functioning code for both sorting algorithms, but you can comment out the lines in your main method which actually do the sorting (so I can uncomment a single line to switch the sorting algorithm). If you'd like, you can ask the user which sorting algorithm and do it that way.

You must also have a function `confirmSorted()`. This function should take your array of numbers, verify that it is sorted in ascending order, and return a Boolean. If the numbers are confirmed to be sorted, print out "Confirmed sorted" and if they aren't sorted, print out "Confirmed NOT sorted". Do this before and after the sort.

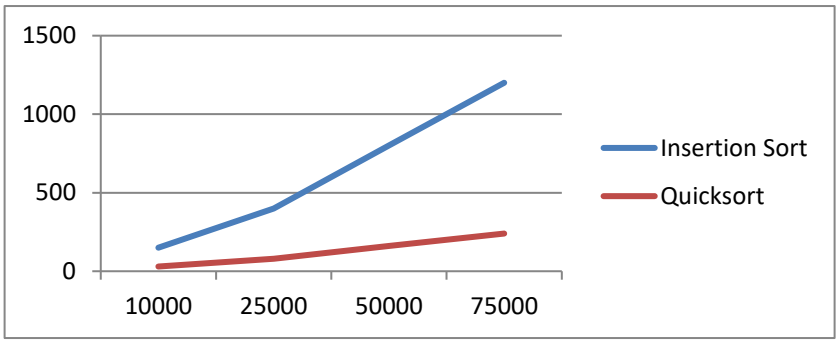
Data analysis

In addition to your code, you'll be asked to try out each sorting algorithm will datasets of different sizes and record the running time (in milliseconds). With this data, you should create a table and graph like the one below. In a paragraph or two, please write what you've discovered with this data, explain what is happening with the data and why.

Example Table Data

Size	Insertion Sort	Quicksort
10000	150	30
25000	400	80
50000	800	160
75000	1200	240

Example Graph



You will be provided with datasets of the following sizes: 10, 100, 1,000, 5,000, 10,000, 25,000, 50,000, 75,000, 100,000, 250,000, 500,000, 750,000, 1,000,000, and 10,000,000. Please include results for all these datasets in your analysis. Some combinations of algorithm and dataset size won't be feasible and may be excluded (the slower algorithm will start to take a VERY long time with large datasets).

Sample Output

Below is an idea of what kind of output your program should have.

```
Reading data from '100000.txt'.
Confirmed NOT sorted.
Sorting using Quicksort.
It took 150 ms.
Confirmed Sorted.
```

Getting Elapsed Time

All programming languages have ways to get elapsed time in milliseconds. The basic idea is to get the time (in milliseconds) before sorting, after sorting, and then find the difference between them. This can vary from language to language, so feel free to ask for help if you need it.

Programming Languages

You may use any programming language as long as you confirm it with me first. This course will be centered on Java so that is my recommendation.

Grading

The assignment is worth 100 points. You will be graded on the following criteria:

Header comment block with name, class, date, project	10
$O(N^2)$ sorting algorithm sorts numbers correctly	20
$O(nlogn)$ sorting algorithm sorts numbers correctly	30
$O(nlogn)$ sorting algorithm meets requirements	10
The confirmSorted() method does what it should	10
Your data analysis is completed and conclusions are logical	20
Total	**100**

If your code doesn't compile, there is an immediate 30 point penalty.

Submitting your work

Your work should be submitted as an archive (.zip, .7z, .rar) on Moodle. This archive must contain all your source code files, as well as your data analysis document. The document can be in any reasonable file format (PDF, .doc, .docx, .odt). Please do not submit your entire project folder.

This project is due Thursday, February 15 at 11:59pm.