Programming Assignment 4

Due: 6 Dec 2024

CSC311, Data Structures

In this programming assignment, you are to implement 5 sorting algorithms and compare their perfor-mance. This assignment consists of two parts: 1) implement and test the sorting algorithms; 2) analyze the running time of the algorithms.

Objectives:
The main objectives of this assignment are as follows:

- Implementing and testing 5 sorting algorithms that we have covered so far.
- Improve your understanding of sorting algorithms and big-O notation.
- Better comprehension of $O(n^2)$ vs. $O(nlogn)$ time.

Getting Started:
Download and import the zip file: PA4.zip for Eclipse:

1. Open Eclipse
2. Select File, then Import
3. Select General category
4. Select Existing Projects into Workspace, and then click on Next
5. Select the button for "Select Archive File" and browse to/select the zip file.
6. Click on Finish. A new project List has been added to the workspace.

Part 1 (Implementation):
Implement three sorting algorithms first and then run the provided time testing code to compare the performance of the algorithms. Here is the list of files that have unimplemented methods that you need to finish:

1. InPlaceSelectionSort.java
2. InPlaceInsertionSort.java
3. InPlaceHeapSort.java
4. MergeSort.java
5. QuickSort.java

6. Test files (using Junit) for InPlaceSelectionSort, InPlaceInsertionSort, InPlaceHeapSort, MergeSort.java, and QuickSort.java. A sample test file for Select Sort InPlaceSelectionSort.java has been created for you. Feel free to create more test cases as you see fit. To test other sorting algorithms, you can copy that file, then modify the code to instantiate a different sort in method setup()).

Important Note:

- Don't cast the generic type E to int in your code. We will test with data other than int type.

Part 2 (Running Time Analysis):
This part of the assignment is provided to help you analyze the performance of the aforementioned algorithms. The required code is provided for you. Just use it!

After implementing the sorting algorithms and testing them with small array, you could test how long it will take to sort random numbers. The time testing code has been already implemented in SortTiming.java. It will take somewhere between 10-15 minutes (although it depends on your machine) to finish all time testing.

Read the code carefully and understand what it does. You might want to run a certain type of sort first, not all of them at once. You can do this by commenting and uncommenting part of the codes. Then run the main method. You will see the progress of the code as it runs through all 5 sorting algorithms for different data size, from 1,000 to 100,000 elements. It measures the time in million second for each sort, and output the result to the console and to a file called output.txt.

After the running is complete, load the data in output.txt to Excel or other tools to plot the following chart. You need to save the data from output.txt and the two graphs in ONE pdf file. Create a line graph of the time testing for all 5 sorting algorithms :

- The horizontal (x) axis should be "input size".
- The vertical axis (y) should be "time in ms".
- The title should be "Running Time of 5 Sorting Algorithms".

Create a line graph of the time testing for the three fast sorting algorithms: MergeSort, QuickSort, InPlaceHeapSort :

- The horizontal (x) axis should be "input size".
- The vertical axis (y) should be "time in ms".
- The title should be "Running Time of 3 Fast Sorting Algorithms".

Submission:
Export the project and save it as PA3.zip (ONLY ZIP file will be accepted not RAR or any other formats). After the export is finished, import your file (i.e., PA5) to Eclipse (if you use other IDEs, you still need to test your project with Eclipse) as illustrated above to make sure your project can be compiled properly.

If the project does not compile in Eclipse, it may not be compiled/executed by the auto grader as well. When the project does not compile, you will get 0 point. In addition to your code, you should submit the running times analysis in one pdf file (i.e., Running Time of Sorting Algorithms). Therefore, two files should be submitted: 1)PA5.zip 2)Running Time of Sorting Algorithms.pdf.

Grading Criteria: (100 pts.)

- Sorting correctly: 70 points.
- Running Time Analysis: 30 points.

FAQ:

- How to fix stack overflow error? Change the eclipse java virtual machine stack size Run -> Run configuration -> Auguments -> VM arguments add -$Xss12800k$
- What does this <E extends Comparable<E» mean? It means that E is not any type, but a type that extends Comparable.
- How to compare two objects that extends Comparable? Use compareTo method E1.compare(E2).
- How to create an array of generic type <E extends Comparable<E» ? (E[]) new Comparable[size]