COP3530 – Assignment 3

Objective

Students will be able to conduct a comparison study of the performance of searching algorithms, by implementing an experiment to compare the running times of well-known searching algorithms.

Assignment Problem

Design and implement an experiment to compare in real time the running times of the following searching algorithms:

- binary search
- sequential search
- sorted search

An implementation of quicksort is given in a separate file (you are required to use it to sort the data in the sorted and binary searches). Time the searching algorithms several times each, and save the results in a .csv file that can be open in an Excel file later. Format your .csv file as follows:

<value of n>, < binary search time>, <sequential search time>, < sorted search time >
<value of n>, < binary search time>, <sequential search time>, < sorted search time >
<value of n>, < binary search time>, <sequential search time>, < sorted search time >
<value of n>, < binary search time>, <sequential search time>, < sorted search time >
<value of n>, < binary search time>, <sequential search time>, < sorted search time >
…

For example (the numbers are not taken from a real example; they are offered to illustrate the content of the file)

100	165448	5553	85531
200	635102	6291	170288
300	1475774	9324	60707
400	457126	12153	63218
500	626482	18096	92991
…			

All of the data (array values and search values) should be randomly generated using the method nextInt() from the java.util.Random class. Time not less than 5000 runs of these algorithms (i.e. your .csv file should have, at least, that number of lines). To time your code use System.nanoTime().

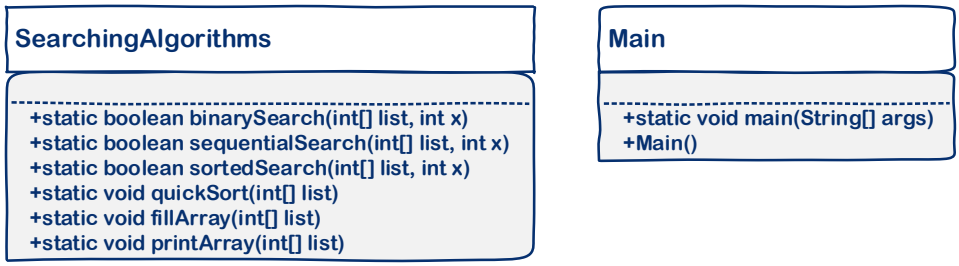
Use Excel to depict graphically the results of your experiment.

What did you observe?

Guidelines

The assignment is to be completed individually or in teams of two students. The given problem is based on the content studied in class on searching algorithms. You are allowed to use all of the code discussed in the lectures. In those cases, make sure you properly credit its source.

Design
Students are expected to structure the code as indicated in the UML class diagram:



Deliverables:
A compressed folder, *PID Assignment 3* (e.g. *1234567 Assignment 3*), containing

- all of the source code of the exercise (the .java files; do not include other files or folders generated by the IDE)
- a document explaining what you observed in the experiment and your conclusions. This document will include the text of your explanations and the picture(s), chart(s), or diagram(s) obtained in Excel.
- the .csv file
- the Excel file

Except for the .csv and the Excel documents, make sure you write your Panther ID as the first line of each file you submit (given as a comment in the Java files). **Do not include your name** in any of the deliverables.

Grading Rubric

The assignment is worth 115 points (out of 1000 total course points). Grade components:

<table>
<tr><th>Component</th><th>Points</th><th>Description</th></tr>
<tr><td>Submission</td><td>5</td><td>The student has submitted the project solution using the requirements for deliverables specified in the Deliverables section.</td></tr>
<tr><td>Organization</td><td>5</td><td>Code is expected to be neat, organized, and readable.</td></tr>
<tr><td>Content</td><td>105</td><td>
<table>
<tr><th>Deliverable</th><th>Points</th></tr>
<tr><td>source code</td><td>65 pts</td></tr>
<tr><td>conclusions</td><td>20 pts</td></tr>
<tr><td>.csv file</td><td>10 pts</td></tr>
<tr><td>Excel file</td><td>10 pts</td></tr>
</table>
</td></tr>
</table>