Raw performance

Efficiency of advanced sorting algorithms

During previous assignments a lot of students show a preference to use build in ADT's like ArrayList and LinkedList, even when using an array is more than sufficient to get the code working. By itself there is nothing wrong with using classes that are provided by Java. But there is always a down-side. During this assignment you and your partner will have to implement quicksort using a LinkedList and with an ordinary array and analyze the time needed for sorting a list and an array containing identical elements and describe the influence of using a LinkedList and ordinary arrays.

Provided code

The code provided on the VLO (`Assignment-1.zip`) contains some classes to get you started. One class, `Player`, might look familiar. This time it implements the `Comparable` interface. The implementation is however not finished. The code contains the details for the implementation.
There is two test classes, one class, `AdvancedSortingTest`, that is used to get the required measurements that are needed for determining the efficiency of the implementations. The other test class is missing some tests that ensure the sorting is implemented correctly.
The class `AdvancedSorts` contains two methods that you have to write the implementation for.

Implementing quicksort

You have to implement two different versions of the quicksort algorithm.

Array version

When implementing this version you are only allowed to use ordinary arrays. The book contains most likely example code for this version. You are allowed of course to add extra (private) methods if you need to. Under no circumstance you are allowed to transform the array to another datatype or use existing code available in the JRE or SDK.

LinkedList version

When implementing this version you are only allowed to use only the `LinkedList` class when constructing lists. Stay as close as possible to the algorithm as shown on slide 27 of the presentation from week 1. When selecting the pivot you can use the first element (head) from the linked-list. Once the pivot has been selected the remainder of the linked-list should be divided into two lists, one containing all elements that are smaller than the pivot, and one containing all elements equal or greater than the pivot. These lists need to be sorted and concatenated in the correct order.

Testing your implementations

Write some tests that assert that your implementation works correct. Add them to the class `ExtendedAdvancedSortingTest`.

Reporting

Your report must conform to the requirements as specified in the study manual (studie handleiding) which can be found on the VLO.
Concretely you must explain your implementations. At least the two sorting implementations and the `compareTo` method from `Player`.

Efficiency

Determine the time-efficiency of both implementations. Please remember that you have to provide some mathematical evidence for the Big-O. (Arguments like '…as can be seen…', '…are similar like…' or '…compared with the figure from bigocheatsheet.com…' are not considered to be valid.)
Is there a difference in efficiency? Is there a difference in raw speed?

Comparison

Compare both your implementation to each other by first describe some quality properties that you find important. Then rate your two implementations to those properties.
Which implementation would you prefer based on the speed and which would you prefer based on your quality properties?

Thought experiment

What do you think would change (raw speed, efficiency) if you used `ArrayLists` instead of `LinkedLists`.

Possible improvements

On slides 34 and 35 a couple of possible improvements are mentioned. Which one could be used (and actually have a positive effect) on the different implementations? Consider implementations based on arrays, `LinkedList` and `ArrayList`. Remember to explain WHY you think an improvement is applicable and why not.

Has this assignment changed your view on ADT's?

As the title says, describe if and how this assignment changed your view on arrays and ADT's, like `LinkedList` and `ArrayList`?

Deliverables

Upload to Moodle a ZIP-file containing your report in PDF-format and all the Java source-code (preferably in the default Maven layout).

Grading

In order to get a positive grading for this assignment your report must meet the requirements as shown in the table below.

Conforms to the requirements as mentioned in the study-manual
Implement the two version of quicksort
Added relevant tests that check for proper implementation
Present the timing measurements using a single table
Present the timing measurements using a single graph.
Correct mathematical determination of the efficiency
Proper analysis of the effects of using an `ArrayList` instead of a `LinkedList` on both the raw speed and the efficiency.
Proper analysis on the applicability of the proposed improvements
Relevant argumentations for view on array vs ADT's