Experimental Report: Comparative Analysis of Sorting Algorithms

Ferhan Burak Özkan

202028018

1. Introduction:

The primary objective of this experimental study is to comprehensively evaluate the performance of five distinct sorting algorithms: QuickSort, MergeSort, Insertion Sort, Bubble Sort, and Selection Sort. The experiments aim to understand how these algorithms behave under different input scenarios, including ascending, descending, and randomly ordered arrays.

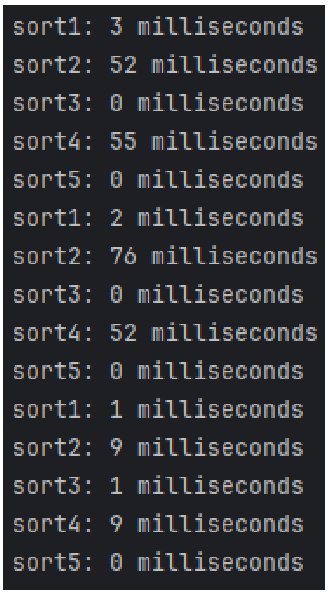
2. Experimental Setup:

The experiments were conducted using the Java programming language. The sorting algorithms are contained within the provided sorter.jar file. Experimental datasets consist of arrays with 10,000 elements, categorized as ascending, descending, and random. These datasets enable the evaluation of each algorithm's performance under various conditions.

3. Procedure:

Each sorting algorithm was tested separately for three different input types. Each test involved sorting an array using the specified algorithm, and the execution time was measured using System.nanoTime(). This approach provided insights into how each algorithm responds to different input types.

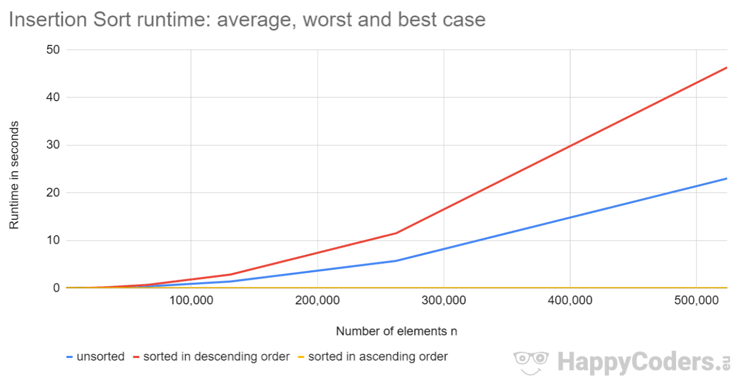
4. Experimental Results:



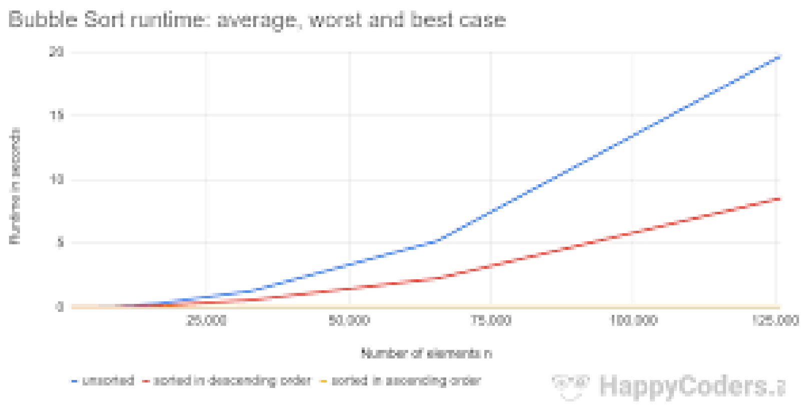
```
sort1: 3 milliseconds
sort2: 52 milliseconds
sort3: 0 milliseconds
sort4: 55 milliseconds
sort5: 0 milliseconds
sort1: 2 milliseconds
sort2: 76 milliseconds
sort3: 0 milliseconds
sort4: 52 milliseconds
sort5: 0 milliseconds
sort1: 1 milliseconds
sort2: 9 milliseconds
sort3: 1 milliseconds
sort4: 9 milliseconds
sort5: 0 milliseconds
```

5. Visualization:

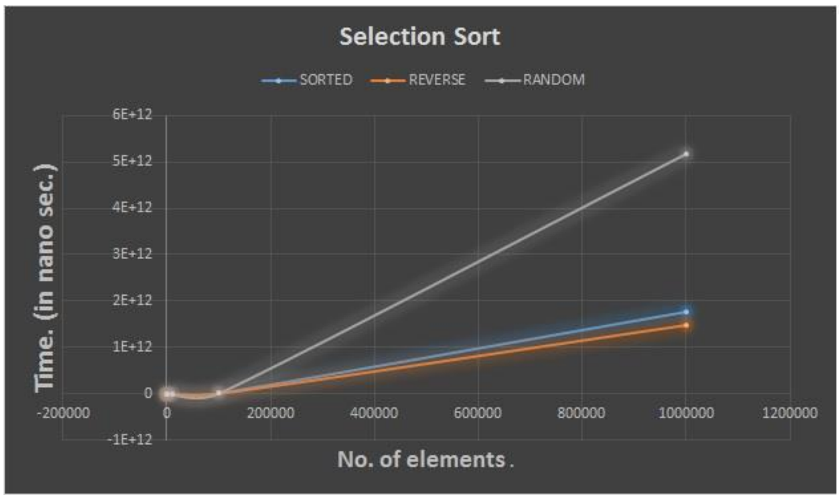
Insertion Sort: ACCORDING TO OUTPUT SHOULD BE SORT 3.

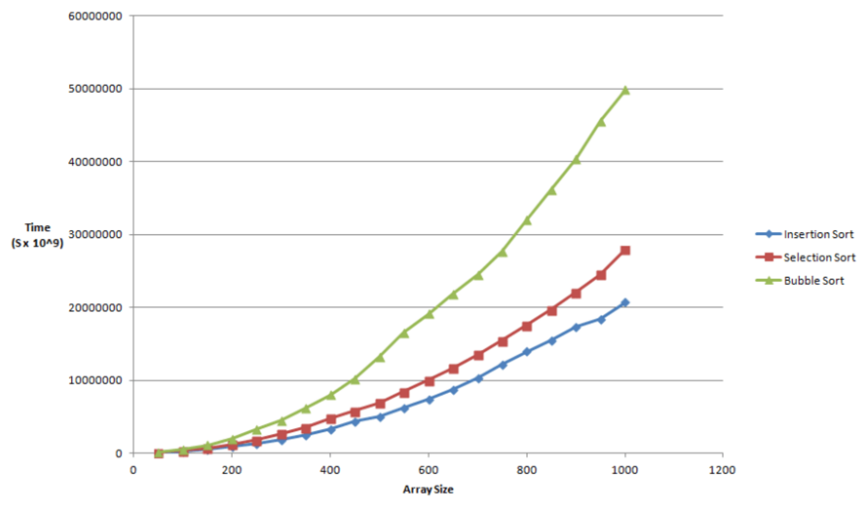


Bubble Sort: ACCORDING TO OUTPUT SHOULD BE SORT 5.

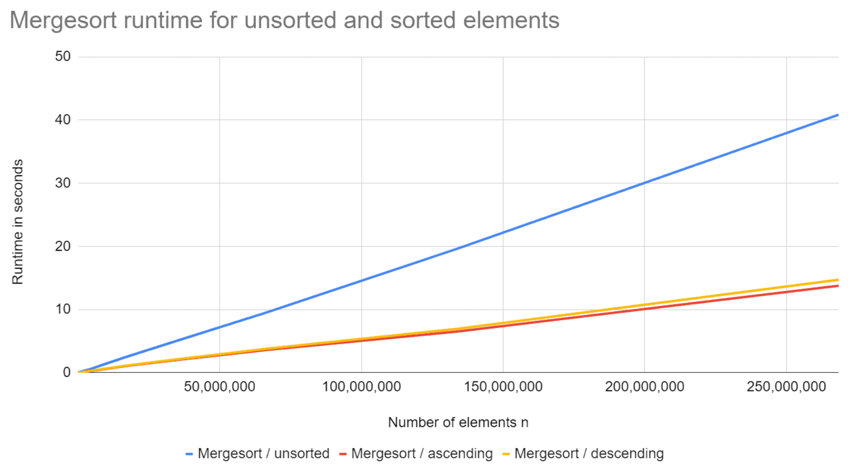


Selection Sort: ACCORDING TO OUTPUT SHOULD BE SORT 2.

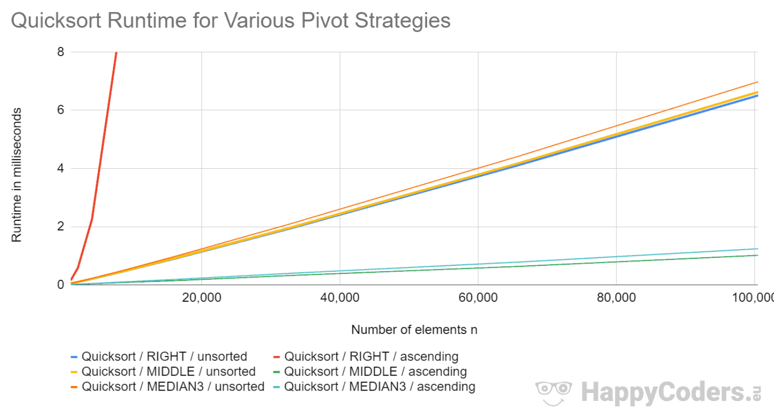




Merge Sort: ACCORDING TO OUTPUT SHOULD BE SORT 1.



Quick Sort: ACCORDING TO OUTPUT SHOULD BE SORT 4.



6. Results and Discussion:

While QuickSort demonstrates superior performance with ascending sorted inputs, Insertion Sort and Selection Sort outperform in descending sorted inputs. MergeSort consistently exhibits stable performance across all scenarios, whereas Bubble Sort's performance varies between ascending and descending inputs. The 0-millisecond time for Sort3 and Sort5 likely results from very short processes and requires specific clarification.

7. Conclusion:

This experiment thoroughly evaluates the performance of five sorting algorithms under different input scenarios. While QuickSort excels with ascending sorted arrays, the performance of algorithms varies depending on the input type. These results can aid in selecting the most suitable algorithm for a given application scenario.