Project 1 Report

Class:
CSE 5311 - 008

Students:
Name: Aadhitya Kumar Student ID: 1001918897
Name: Kenil Patel Student ID: 1002332816

Sources Used:
- Official Java Documentation: https://docs.oracle.com/javase/8/docs/api/
- CSE 5311 slides

Description:
Aadhitya Kumar has completed the Insertion Sort and Quick Sort coding part using official Java documentation, and Kenil Patel (Used the Java class summary from Oracle for reference) has completed the Generation of datasets and Merge Sort part. The rest of the part, including the calculation of time complexity, comparison of time taken for all three algorithms, experimental and theoretical results for the report, is completed collectively.

Time complexity of Insertion Sort:
Best case: T(n) = $\Theta(n)$
Worst case: T(n) = $\Theta(n^2)$
Average case: T(n) = $\Theta(n^2)$

Experimental results for Insertion Sort:
For 40 students: 11 microseconds
For 1000 students: 3591 microseconds
For 6000 students: 11226 microseconds

Time complexity of Quick Sort:
Best case: T(n) = $\Theta(n \lg n)$
Worst case: T(n) = $\Theta(n^2)$
Average case: T(n) = $\Theta(n \lg n)$

Experimental results for Quick Sort:
For 40 students: 18 microseconds
For 1000 students: 586 microseconds
For 6000 students: 1244 microseconds

Time complexity of a Merge Sort :
Best Case : $\Theta(n \lg n)$
Average Case : $\Theta(n \lg n)$
Worst Case : $\Theta(n \lg n)$

Experimental results for Merge Sort:
For 40 students: 31 microseconds
For 1000 students: 897 microseconds
For 6000 students: 1829 microseconds

Experimental and Theoretical results:
The theoretical results give us bounds on the running time of each algorithm. The experimental results represent how much real world time each algorithm takes. The experimental results appear to be within the bounds given by the theoretical results.

Comparing and Contrasting the Three Algorithms:
For 40 students, Insertion sort was the fastest at 10 microseconds. This is followed closely by Quick Sort at 13 microseconds, with Merge Sort far behind at 32 microseconds. This is a bit of an anomaly since insertion sort is asymptotically slower than both on average. The reason Insertion sort and Quick Sort are much faster on smaller inputs is likely due to Merge sort requiring extra memory overhead. The reason that Insertion sort was faster than Quick sort on this input is likely due to the input being closer to Insertion sort's best case, which is asymptotically faster than Quick sort's. For 1000 students, Quick sort was the fastest at 586 microseconds, Merge sort was the second fastest at 897 microseconds, and Insertion sort lagged far behind at 3591 microseconds. Due to the larger input, the running times are now much less dependent on memory overhead and more reliant on their respective time complexities. It is also less likely that an algorithm will receive an input close to its best case. A similar trend emerges with 6000 students, where Insertion sort is even further behind Merge sort and Quick sort. Quick sort still has an advantage over merge sort due to using less memory. One small difference in sorting is that if a duplicate value is found, Quick sort puts it before the first instance of it while Merge sort and Insertion sort put it after.

Honor Code:

I pledge, on my honor, to uphold UT Arlington's tradition of academic integrity, a tradition that values hard work and honest effort in the pursuit of academic excellence.

I promise that I will submit only work that I personally create or that I contribute to group collaborations, and I will appropriately reference any work from other sources. I will follow the highest standards of integrity and uphold the spirit of the Honor code.

I will not participate in any form of cheating/sharing the questions/solutions.

Aadhitya Kumar Aadhitya Kumar 10/3/2025

* 5311 - Design and analysis of Algorithms.

⇒ Project - 1

⇒ HONOR CODE :

I pledge, on my honor, to uphold UT Arlington's tradition of academic integrity, a tradition that values hard work and honest effort in the persuit of academic excellence.

I promise that I will submit only work that I personally create or that I contribute to group collaborations, and I will appropriately reference any work from other sources. I will follow the highest standards of integrity and upload the spirit of the Honor Code.

I will not participate in any form of cheating/sharing the questions/solutions.

Signature ~~Kpatel~~

Date

10/03/2025.

(Kenil Patel).