**Compare and Contrast Execution Time**

In this lab, three different sorting algorithms (bubble sort, insertion sort, and merge sort) were exposed and used in the SortAlgorithm library. The functions and the class were written in C++ natively (comparison application) and exposed as a library using emscripten to be used with Javascript and HTML (demo application). The execution times of the library in comparison application were measured with high resolution clock offered by standard chrono and the demo application had its library execution time measured with performance.now(). Both were measured in microseconds. Each respective codebase was run over 20 iterations and each sorting algorithm sorted 1000 integer elements.

**Bubble Sort Timing Comparison**

The average execution time for bubble sort in comparison application was approximately 8,087 microseconds and at 95% confidence interval, the margin of error came out to be 2570 microseconds. In the demo application, the average execution time was 89,890 microseconds and the margin of error of 15,100 microseconds at 95% confidence interval. The average execution time for the bubble sort in the demo application was around 11 times slower than its native counterpart. As it can be seen visually, the margin of error for the demo application was significantly higher, and using 95% confidence interval, it was found that 95% of data rests between 74,800 and 105,000 microseconds. These results are statistically significant as even though there exists some variance in data, at 95% confidence interval, the execution times between the demo and comparison applications were significantly different, with the demo application being around 11 times slower.

**Insertion Sort Timing Comparison**

The average execution time for insertion sort for the comparison application was around 2456.1 microseconds. At 95% confidence interval, it was found that 95% of the data rested between 2330 and 2580 microseconds with the margin of error being 126 microseconds. For the demo app, the average execution time was around 27755 microseconds. The margin of error was around 7870 microseconds, about 62 times larger than the margin of error for the comparison app. The execution times for the demo application with respect to the insertion sort had much variance and a much broader confidence interval, with 95% of the values resting between 19,900 microseconds and 35,600 microseconds. At 95% confidence interval, the execution times for the demo application and the comparison application are very different. The largest value in the confidence interval of the comparison app was 2,580 microseconds and the smallest value in the demo app within 95% confidence interval was around 19,900 microseconds, being 7 times slower than the comparison app. With this, it can be observed that this comparison of execution times of insertion sort is statistically significant.

**Merge Sort Timing Comparison**

The average execution time of merge sort performed in the comparison app was around 278 microseconds and its demo application counterpart was approximately 870 microseconds. The average execution time for the demo application was only about 3 times slower than that of the comparison application. At 95% confidence interval, the execution times for the comparison app sat between 240 and 317 microseconds while the execution time for the demo application sat between 736 and 1000 microseconds. At their closest numerical data points, the demo application performed around 2.3 times slower than its comparison application showing statistical significance. Both insertion sort and bubble sort had some significant differences in their average execution times between the demo and comparison app with the demo app for both being around 11 times slower than their respective counterparts for comparison applications. Merge sort for the demo application, however, was only around 3 times slower than the comparison application. This may be due to the inherent speed superiority of merge sort compared to the other sorting algorithms.

**Total Execution Time Comparsion**

Finally, a timing comparison of the total execution time of all three algorithms was done for each application. As it can be seen visually, the native comparison app was much faster than the demo application. Numerically, the comparison application was around 10.7 times faster than the demo application, as was the case for most individual algorithms. At 95% confidence interval, the execution time of the comparison app rested between 10,500 and 11,600 microseconds with a margin of error of 573 microseconds while for the demo app, 95% of data sat between 105,000 to 132,000 microseconds with a margin of error of 13,900 microseconds. Just from the visual aspect of the chart alone, it can easily be seen that the data between the two applications do not intersect and thus show statistical significance.