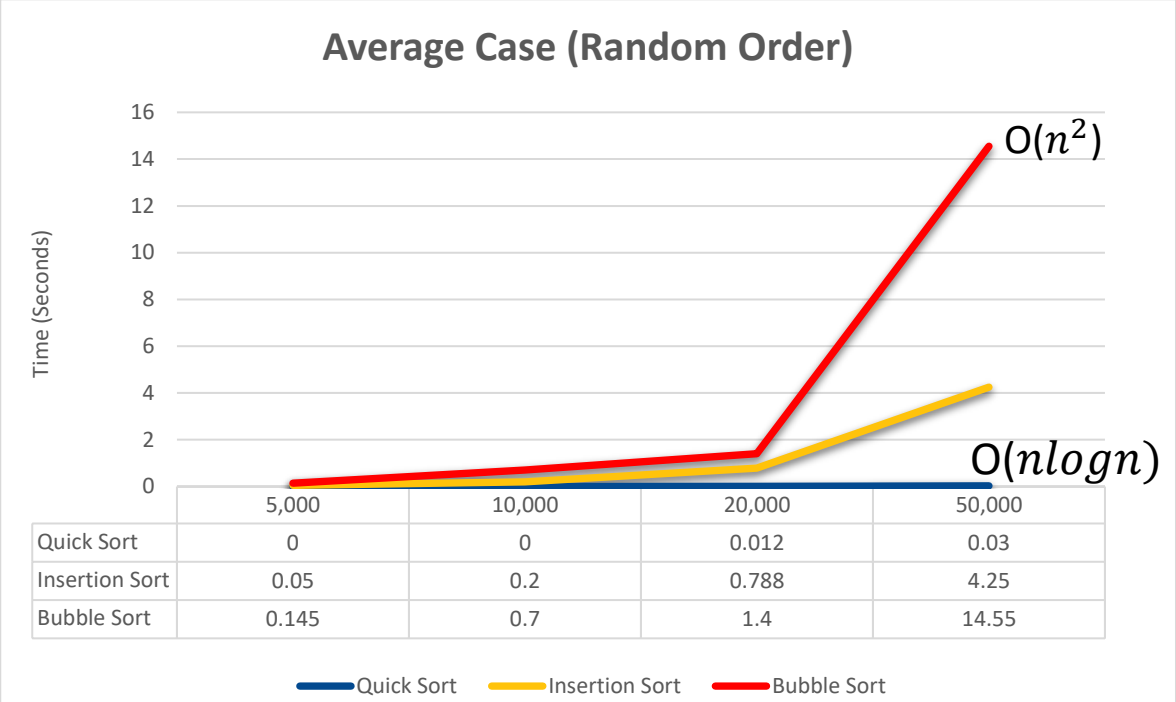
Elba Chimilio
COP3530 - Section 7303

Sorting Algorithms Report



	5,000	10,000	20,000	50,000
Quick Sort	0	0	0.012	0.03
Insertion Sort	0.05	0.2	0.788	4.25
Bubble Sort	0.145	0.7	1.4	14.55

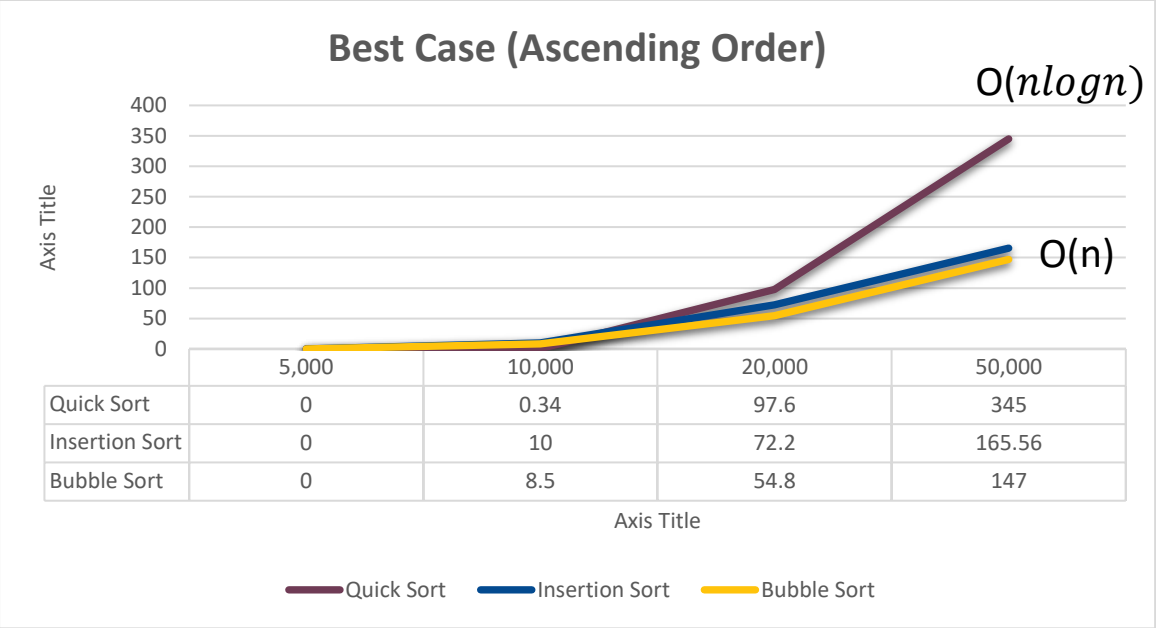
The graph measures the time (in seconds) versus the file size. In the ***average*** case, the file has randomly ordered integers and is used to measured the selected algorithms (quick, insertion, and bubble). I measured it six times and took the average which I then used to display each result.

As shown in the graph, bubble sort has an O(n^2) shape, while insertion sort did not look like the O(n^2). It appears as if it is reaching the shape, but could be limited by the data. On the otherhand, quick sort has not formed its O(nlogn) shape, but again, it could the limited data. The average case does not have a set or sorted amount of integers, which for each of these algorithms does not give desired computational complexities for randomized numbers.

The average case clearly showcases the increase in computational complexity for each case when compared to the best case (ascending order). In the worst case however, the average case and the worst case are almost identical with the exception of quick sort which has an average case of O(nlogn).

Elba Chimilio
COP3530 - Section 7303

Sorting Algorithms Report



	5,000	10,000	20,000	50,000
Quick Sort	0	0.34	97.6	345
Insertion Sort	0	10	72.2	165.56
Bubble Sort	0	8.5	54.8	147

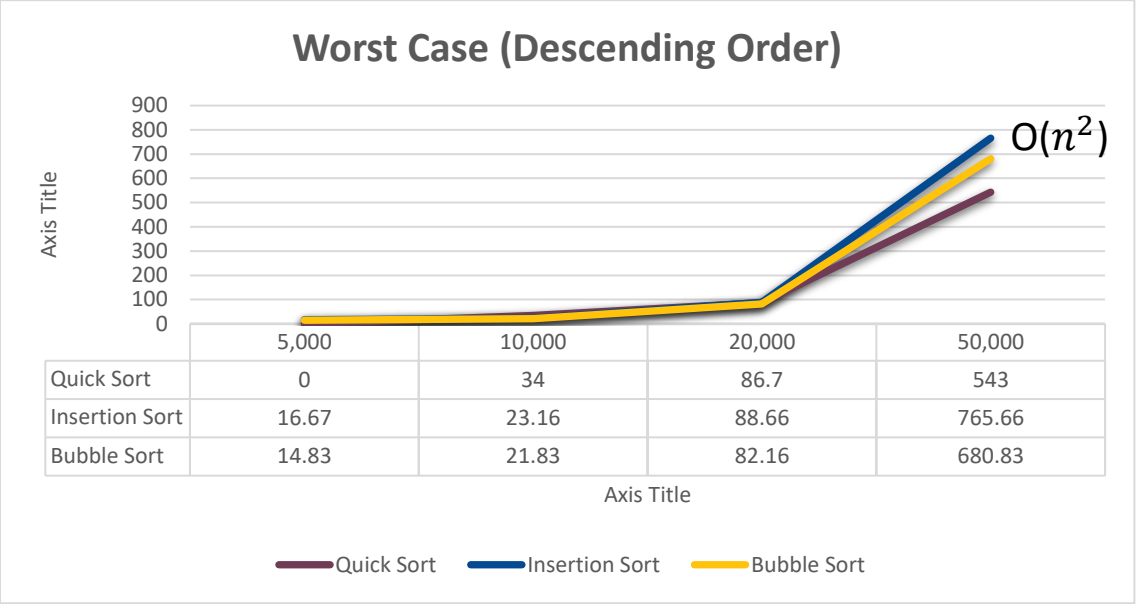
The graph measures the time (in seconds) versus the file size. In the ***best*** case, the file has integers in ascending order and is used to measure the selected algorithms (quick, insertion, and bubble). I measured it six times and took the average which I then used to display each result.

As shown in the graph, bubble sort and insertion sort appear to have an O(n) shape while quick sort has an O(nlogn) shape. The best case includes integers in ascending order, so the bubble sort and insertion sort algorithms have less comparisons to do. Quick sort has an O(nlogn) in both the best and average case. My data does not display an O(nlogn) shape for a quick sort in the average case, but it could be because there is not much data.

The files with ascending order have the overall best results on the computational complexity of each algorithm affecting how they partion, pivot, swap, or compare.

Elba Chimilio
COP3530 - Section 7303

Sorting Algorithms Report



The graph measures the time (in seconds) versus the file size. In the ***worst*** case, the file has integers in descending order and is being measured against the selected algorithms (quick, insertion, and bubble). I measured it six times and took the average which I then used to display each result.

As shown in the graph, all the algorithms seem to be taking on the shape of O(n^2). The worst case has integers in descending order, so each algorithm is effected within the process. For example, bubble sort would have to continously compare each and every pair until the end of the file or in insertion sort, the pivot would be affected the most. Compared to the best case, there is a drastic difference, with each algorithm getting worse with a descending order file. However, in the average case, quick sort is the only difference when comparing the two, as insertion and bubble sort are O(n^2) for both the worst and average cases.