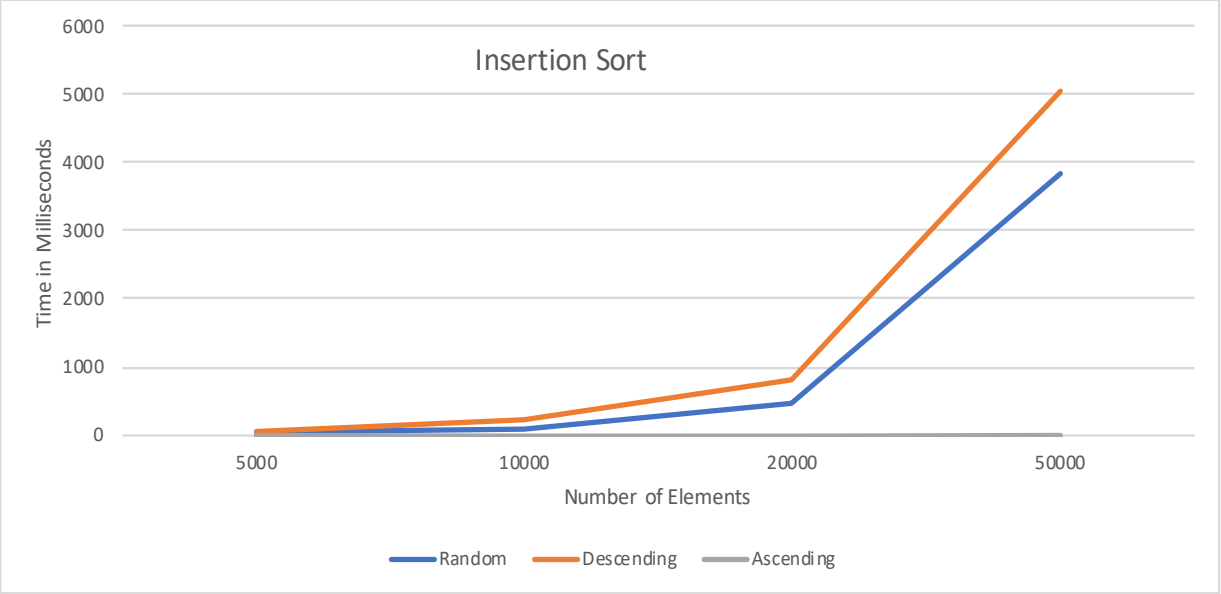
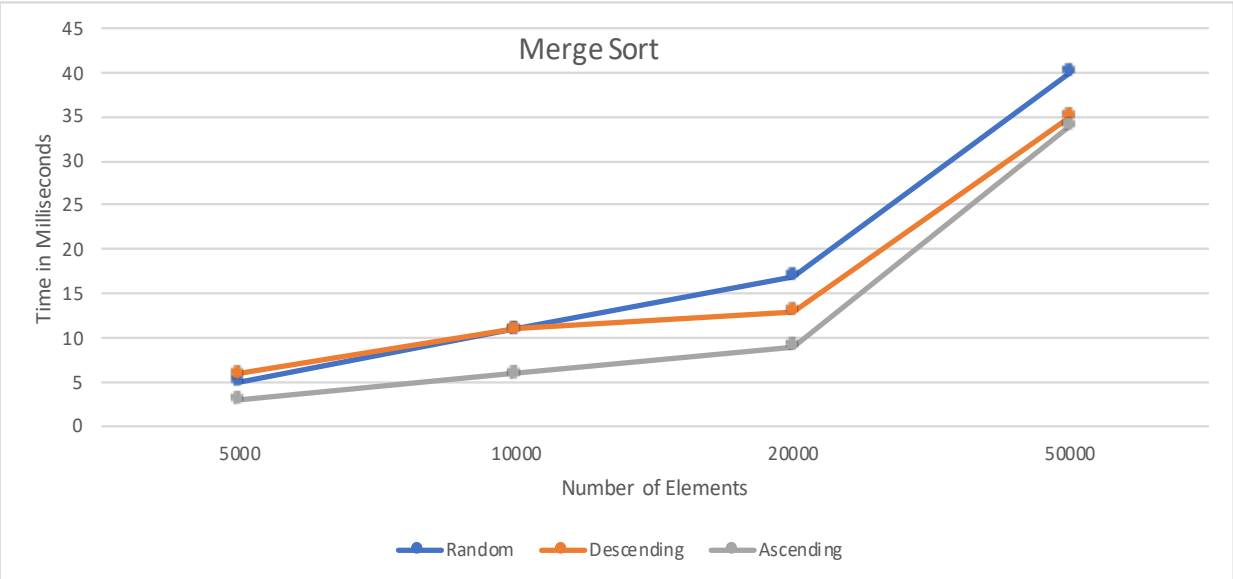
All values are averages from 3 runs of the sorting algorithms.

Insertion Sort				
Elements	5000	10000	20000	50000
Random	50	100	485	3834
Descending	59	224	820	5041
Ascending	0.2	0.5	0.8	1



Merge Sort				
Elements	5000	10000	20000	50000
Random	5	11	17	40
Descending	6	11	13	35
Ascending	3	6	9	34



Aaron Williams
Project 2

By investigating the time complexities of insertion sort and merge sort through the graphs above, it is glaring that the merge sort has a much lower average time than insertion sort for the same input files. The known complexities of the two sorting algorithms are very different, insertion sort is known to take $O(n^2)$ in the worst case and O(n) in the best case while merge sort is O(n log n) for all cases. This is apparent in the best case (ascending order) because the merge sort takes longer than the insertion sort, typically by a large amount. However, in the average and worst cases, the merge sort is exponentially faster. In the worst case with 50,000 elements, merge sort is able to sort the entire list in an average of 35 milliseconds, while the insertion sort takes an average of 5041 milliseconds; essentially, the insertion sort takes 144 times longer than the merge sort in this case. The order of the data in this case, dictating if the algorithm will be best case or worst case (ascending being the best case, and descending being the worst case) and the size of the file dictating the scale of the time (the more data entries, the longer the time).