Lab 5

	O(nlogn) Merge Sort	O(n^2) Insertion Sort
10	?	3
100	?	25
1000	?	1859
10000	?	99124
100000	?	DNF
1000000	?	DNF

For lab 5, I chose to create a merge sort algorithm form my O(n logn) solution, and an insertion sort algorithm for O(n^2). Due to limitations imposed by the Coronavirus outbreak, I was unable to get my merge sort code fully functional (strangely, a print statement in my mere function alters the output significantly - without the "printf("four\n");" on line 121, a set fault is thrown, while with the statement, the code executes, but returns garbage from memory apparently). Based on my findings with insertion sort, however, the issue with exponential algorithms becomes rapidly apparent; larger datasets, such as the 100,000 and 1,000,000, never even completed. Presumably, the merge sort would be much better suited to handle these sizes of datasets. The primary pitfall of the insertion sort is rooted in its nested loops (lines 79 and 82) that are necessary to traverse both the old and new arrays.