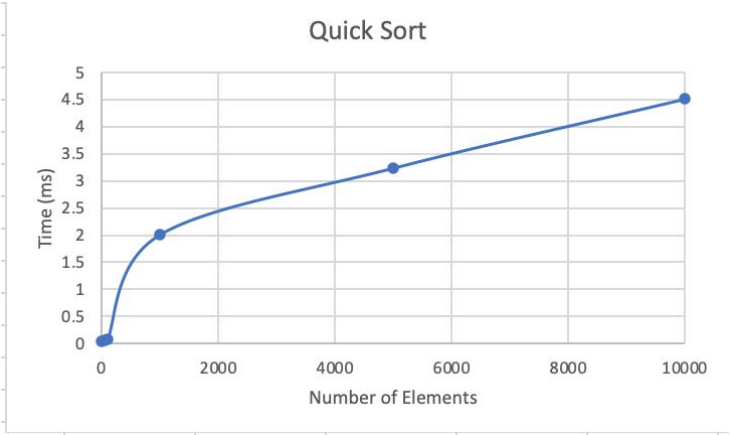
Algorithm Efficiency and Sorting Observations

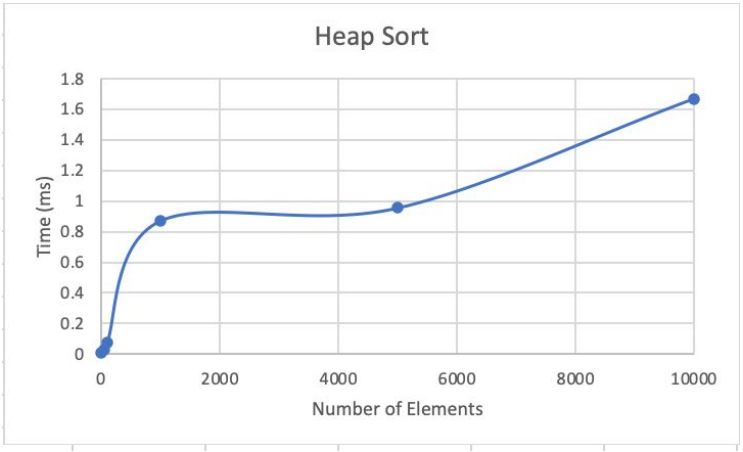
To generate the data for the graphs showing the efficiency for the sorting algorithms, I used random.org generated a set of random numbers for each of the 6 tests each of the numbers being between 1 and 10,000. The time it took to execute the sort is measured in milliseconds and the maximum number of elements in an array measured was 10000. It is difficult to tell if all of the graphs for the sorting functions algorithm efficiency follow closely to the big O notation for the given sorting function for a few reasons. One big O notation measures the number of instructions that must be executed and not the time, two big O notation gives the worst case scenario for each sorting algorithm and the data given was not the worst case. Even so I would say that both Radix sort and Bubble sort closely match their Big O notations with the graph for Radix sort being linear and the Graph for bubble sort being quadratic.

Quick Sort ($O(n^2)$)



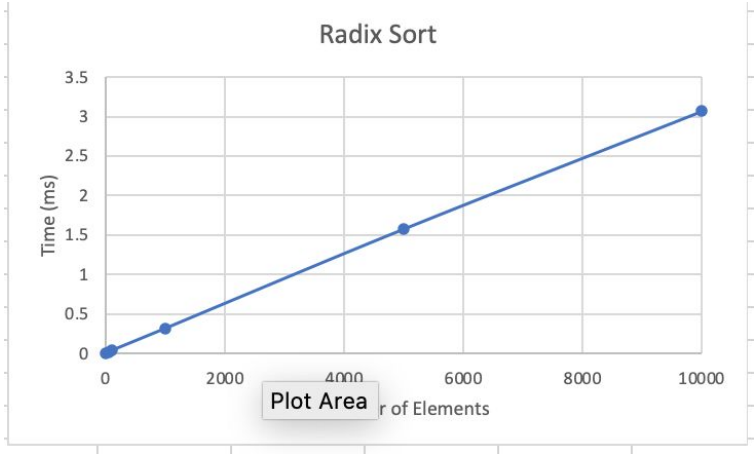
Quick sort was the sorting algorithm that was the third fastest sorting algorithm 4/6 test with it taking the most time for the first two tests consisting of 10 and 50 elements.

Heap Sort (O(nlog(n))



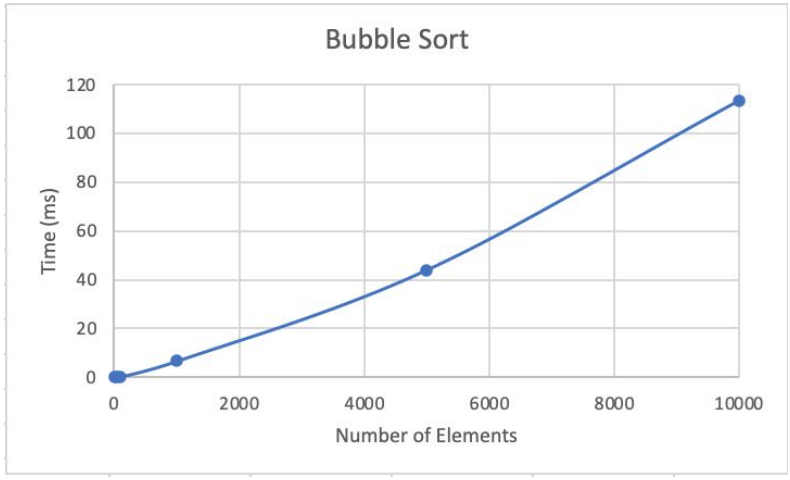
Heap Sort was the sorting algorithm that gave the fastest time for the trials with more than 1000 elements in the array.

Radix Sort (O(n+k))



Radix Sort was the fastest sorting method for the trials with 1000 and less elements, I think the fact that the numbers were limited to 10000 being the maximum had a lot to do with Radix's sorting speed being so fast.

Bubble Sort ($O(n^2)$)



Bubble Sort was consistently the slowest sorting algorithm after the first 2 where quick sort was the slowest.

Quick sort and Heap sort have similar graphs even though Quick Sorts Big O notation indicates that its graphs should be more similar to Bubble Sort as they have the same Big O notation, this may be due to the quick sort functions average case being the same as Heap sorts worst case.