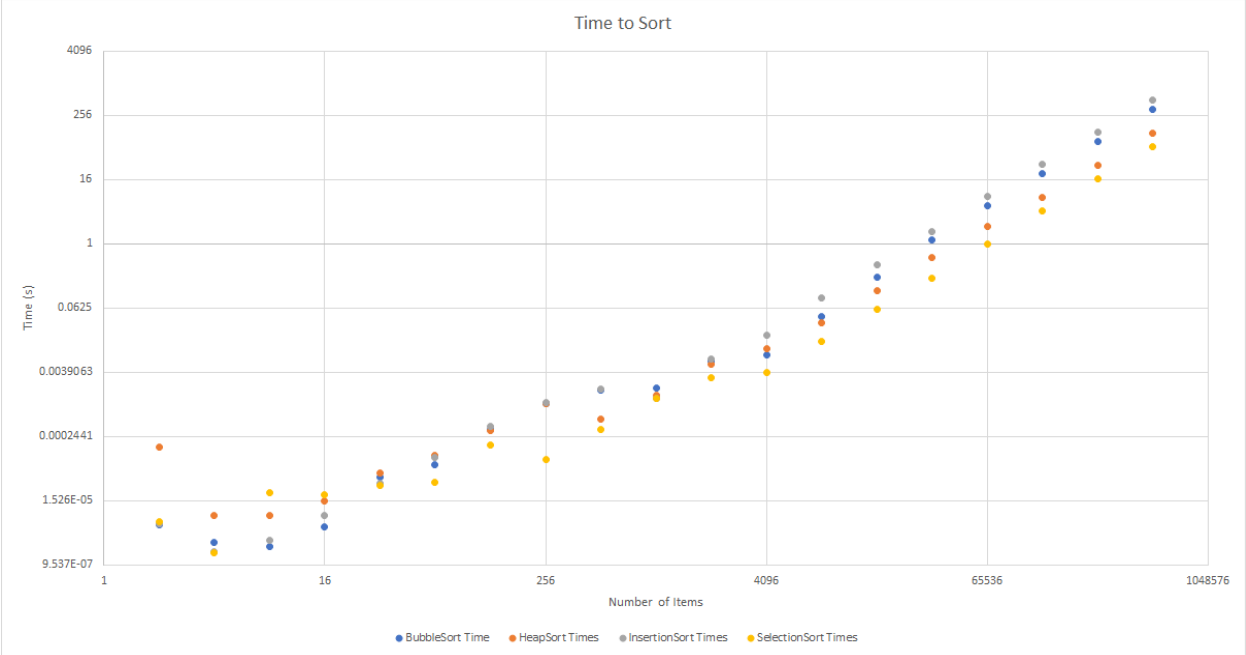
Intro:

The assignment was to write and benchmark four different sorting algorithms in Java: Bubble Sort, Selection Sort, Insertion Sort, and one sort that had a time complexity better than $O(n^2)$. I decided to go with Heap Sort because I think sorting things with heaps sounds kind of cool and also the Geeks for Geeks page I read about said they were made with 'Complete Binary Heaps' and that sounds even cooler!!! Afterwards, we had to graph the performance of each of the sorts. Since our fourth sort had to perform with better time complexity than $O(n^2)$, I predicted that my Heap Sort would look better on the graphs when compared to the rest of the sorts

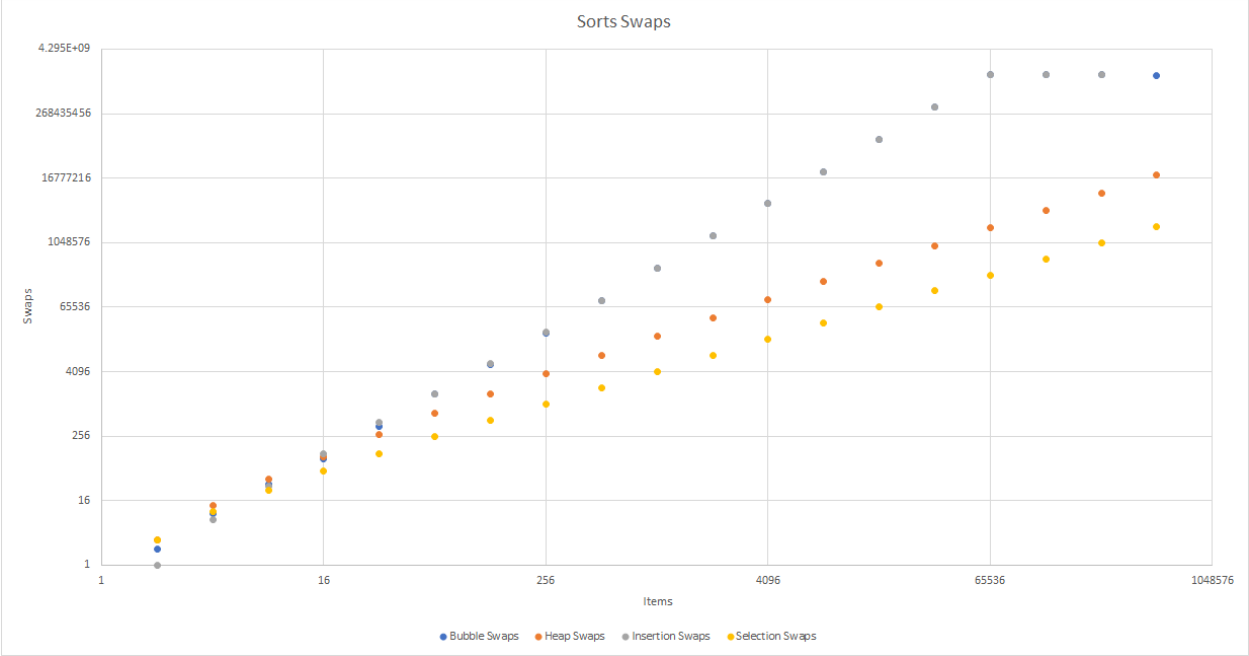
Data:

Time Comparisons:



In order, it looks like the quickest sorts under five minutes were: selection, heap, bubble, and insertion. I think it's easy to see why insertion sort is the slowest operation, as it is a double for-loop and a while-loop for as long as the previous element in an array is greater than the currently selected element. On the same token, heap sort only involves creating a heap (which has a time complexity of N) and removing the root from the heap (which is log(N)). Interestingly, there is a point in both Selection Sort and Heap Sort where sorting a small array, about two elements, is significantly slower compared to an array of about four or eight sizes. There is similar behavior in the other sorts, but I thought it was more noticeable in those two. Heap Sort has a point at the start where it is significantly slower than the others.

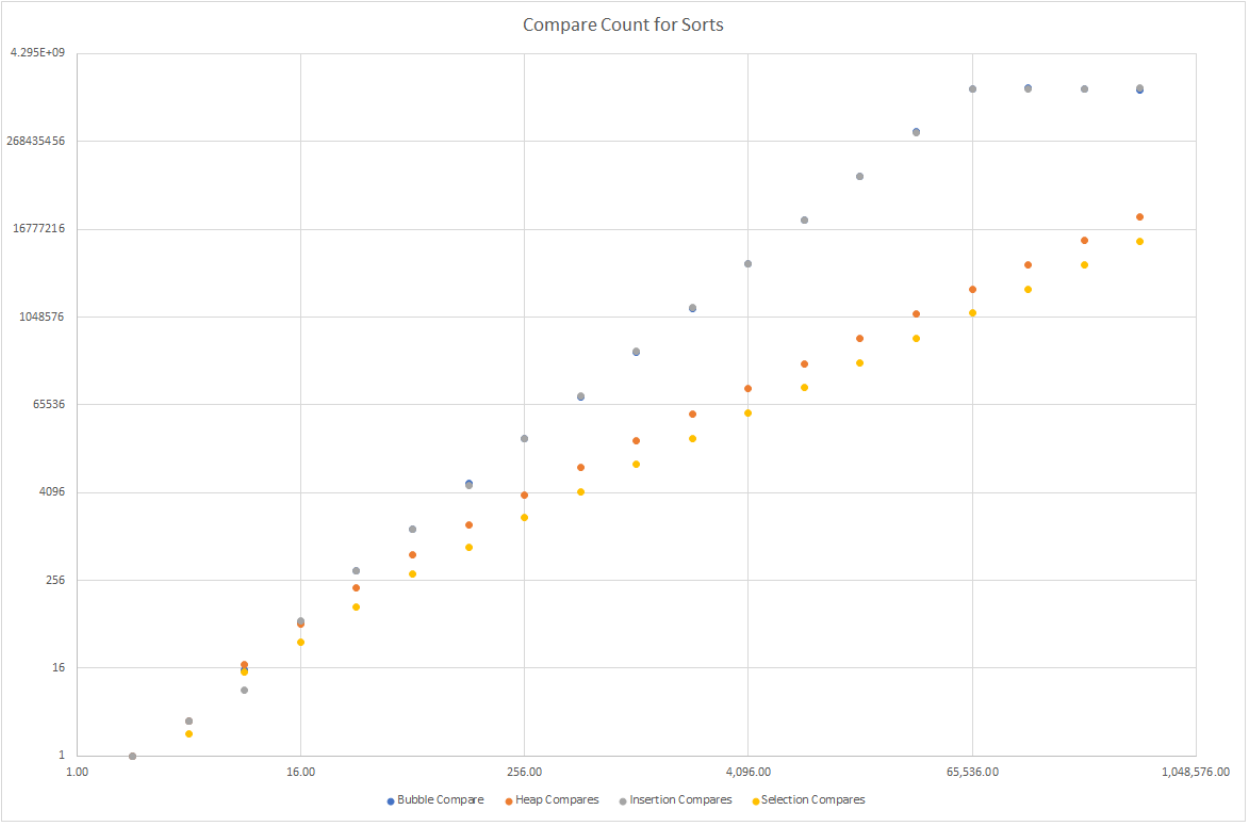
Swap Comparisons



Overall, the slower the sort was in time the more swaps it had to go through. There seems to be a little bit of an exception because Bubble and Insertion have almost an identical number of swaps, but on the time graph Bubble performs a little bit better than Insertion sort. It is also

surprising to me that the difference in the number of swaps between Bubble, Insertion and Heap and Selection gets so large. The time graph doesn't have as large of a gap in it.

Compare Counts:



All sorts seemed like they did a lot of comparing, and so to be honest I didn't really keep track of them to think about which ones would be better later. I think it's interesting that Insertion Sort and Bubble Sort straight up flatten out, though. I checked their time graphs to see if they appeared to flatten out there, too, but they did not. If the sorting times did not take so long when I was graphing them, and if my battery wasn't about to die at the time, I would have investigated that further to see if I could make the lists bigger and see how the compare count continued to behave on the graph. This graph is also similar to the swap counts and that makes sense.

Conclusion:

At first I thought that Heap Sort would perform noticeably better than all of the other sorts, but looking at the graphs it seems like that was not the case. Heap performed similar to Selection in the amount of time it took to sort, but Heap really lagged behind Selection with the number of swaps. The difference was not as bad in comparisons but still noticeable.