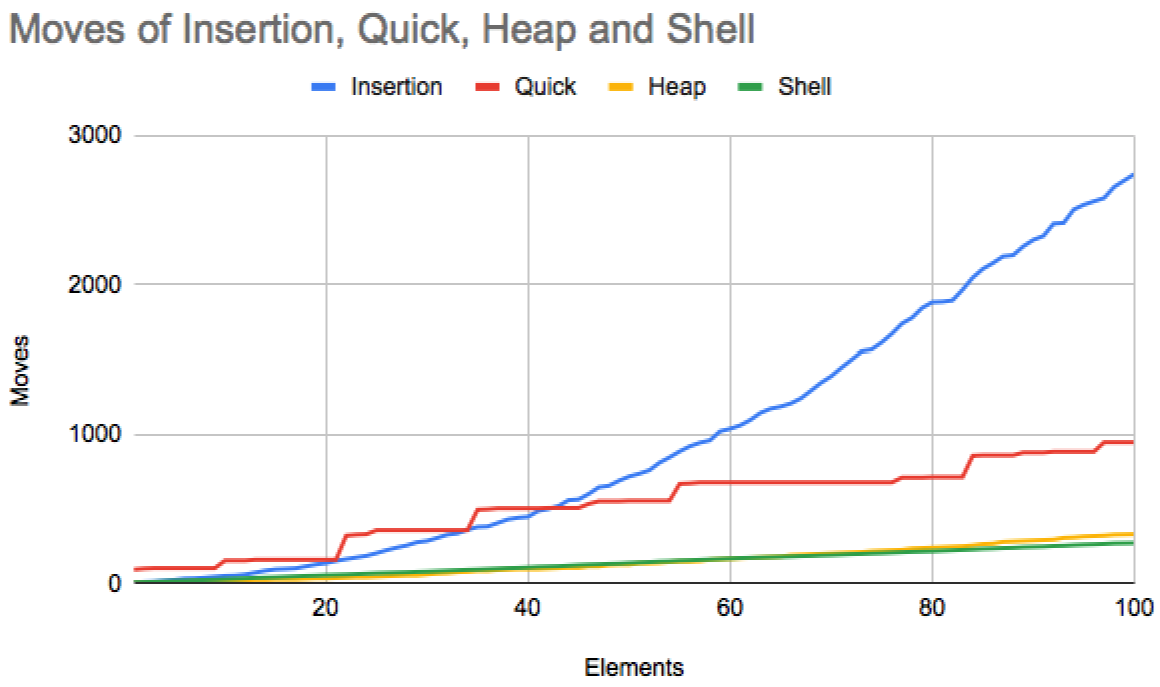
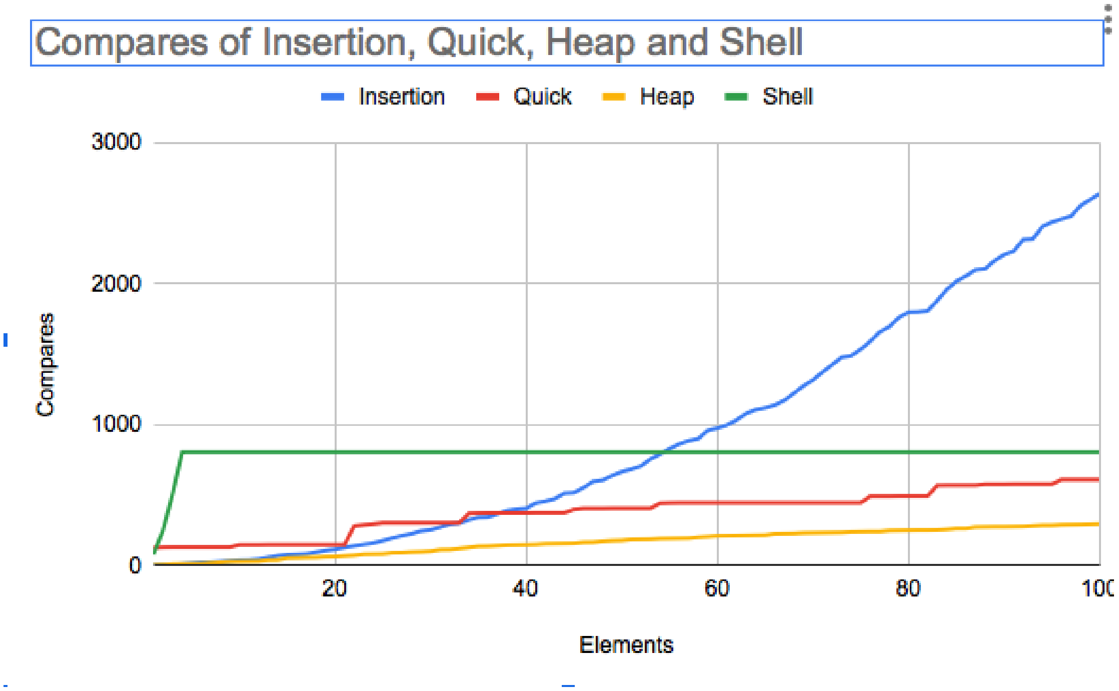
Assignment 3: Sorting

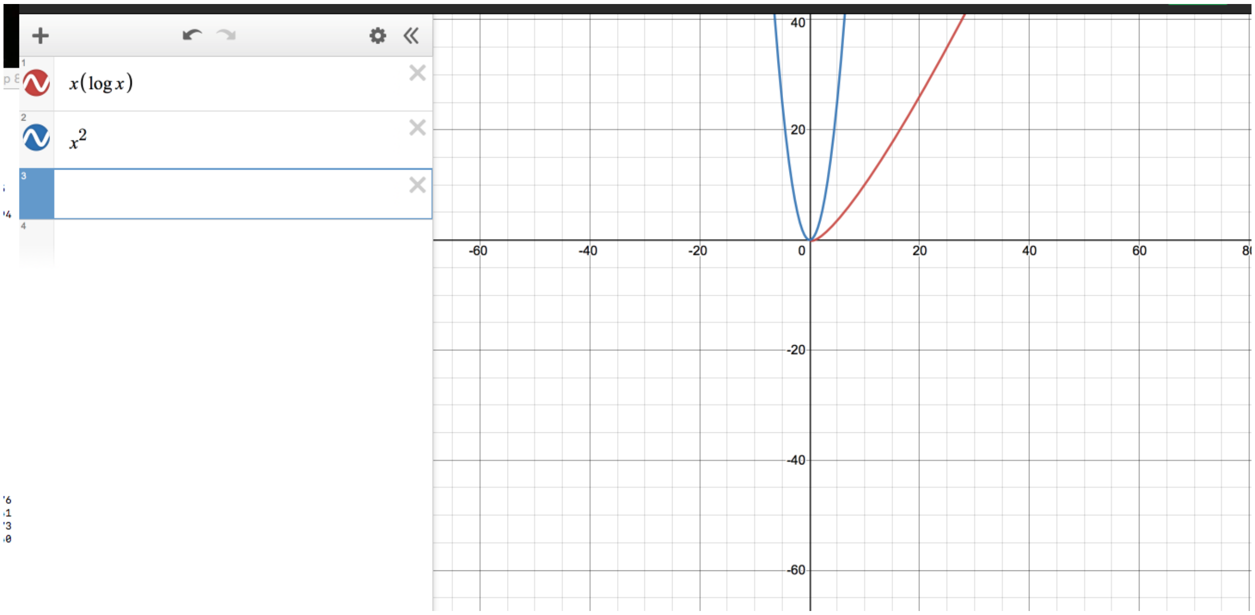
In this assignment we implemented different sorting algorithms, such as Heap, insertion, quick, and shell sort , to keep track of the number of compares and moves of each sorting algorithm.



From this graph of comparing the moves of each sort for the first 100 elements we see how much faster insertion sort is increasing. This means that insertion sort is the least efficient. After that we see the quick sort is the next least efficient. The heap and shell sort are relatively the same efficiency.



Similar to the moves compare, we see that insertion sort is also least efficient as it rapidly increases. We see the shell sort is the next efficient. Finally, heap and quick sort are also relatively similar although heap sort is the most efficient. We see that the comparisons of the shell sort are constant after a certain amount of elements which can be due to my algorithm of comparisons.


1 $x(\log x)$

2 x^2

3

4

From this graph we compared the time complexity of each sort. The time complexity of quick and heap sort is both O(n*log(n)) while the time complexity of insertion and shell sort is both O(n^2).

The time complexity of insertion and shell sort, especially insertion sort, is evident from how fast the graph grows in both the moves and comparisons. We see that the time complexity plays a big factor in the sorting algorithms.

While the graphs of quick and heap sort generally were much slower due to the time complexity of O(n*log(n)). We can see this in the graphs of the comparisons and moves.

In conclusion, this assignment shows the importance of breaking down each algorithm into its own. It came to show the importance that there is no one better algorithm than another. I think this assignment was eye opening to realize the importance of time complexity and efficiency. It also made me realize the importance of having pseudocode beforehand. The pseudocode provided was extremely helpful in completing this assignment and I normally do not plan ahead before I code. After this assignment, I most definitely will plan pseudocode before starting any assignment.