The point of this project was to implement the sorting algorithms Selection, Heap, Merge and QuickSort. We were to test these sorters on random arrays of increasing orders of magnitude to examine their scaling behavior. We graphed our results, extrapolate our results and experimentally verify the accuracy of our estimates. So from the users input from the command line, I had the driver class select the correct sorting class out of the four options and also construct an array of length n = 10^k of random double values, where k is obtained from the command line. The driver printed the name of the sorting algorithm the user chose and some information on arrays constructed. The information included the size of the array, the time it took to finish the sorting, how many times the sorter makes comparisons and how many moves.

Using a combination of sorting, swapping and other methods I was able to get the desired output. Each sorter had its own way of sorting. Some of these classes where very short like selection sort. Selection sort consisted of a sort method and a swap method. On the other hand, Heap sort has a sort, heapify, siftdown, poll, samllerChild and swap method. It is these procedures that makes them unique. In their own way, these sorters sort arrays.

To make sure my program was working correctly, I made a simple verify method in the driver that right after an array is sorted it would double check the array is in ascending order. I also took the liberty to correct the user if they were not typing their inputting correctly in the command line. After that was done, I ran the program multiple times using all four sorters to make sure they were all working correctly.

With this project I was able to compare these sorting algorithms nicely. I got a good idea on how many comparisons the sorter makes, moves it makes and most importantly the time it takes to complete a sorted array. As for predicting, according to my sorting estimate tables, selection sort is the only outcome you can closely predict. The charts show that selection takes the longest time. It seems like for arrays of under 100, selection sort is what you want but not for arrays of size over 1000.