

After running bubble, insertion, selection, merge, and quick sort algorithms on arrays starting at 20,000 elements and doubling until 640,000 elements, I prepared the data in various graphs and charts so it could be easily compared. The time in which it took these five sorting algorithms to sort the various lists was exactly as I had suspected, only because we walked through how bubble, insertion, selection and, merge sort work in class. From the reading I gathered that quick sort would be the fastest of the five. I was surprised at how long bubble sort took to sort an array of 640,000 elements. When I ran the bubble sort algorithm I made sure nothing else was open on my laptop, and it still took roughly 30-45 minutes. During which time I spent time reading. It is also interesting to note that running the bubble, insertion, and selection algorithms on my laptop made the fan spin at a rapid speed, which made the laptop sound as if it were getting ready to take off. Merge sort and quick sort on the other hand both finished in under 500 milliseconds and caused no noticeable decrease in the performance of my laptop.

Looking at the bubble sort line graph a noticeable increase in time occurs once the algorithm begins to sort an array of 80,000 or more elements. The time it takes for bubble sort to sort through arrays consisting of more than 80,000 elements increases exponentially. The same can be said for the insertion sort algorithm although it is almost three times the speed of the bubble sort algorithm. The selection sort algorithm was just slightly slower than the insertion sort algorithm. It sorted the array of 640,000 elements in a time of 575,838 seconds and insertion sort sorted the array in 518,671 milliseconds. Both the merge and quick sort algorithms were way faster than any of the previous sorting algorithms tested. The quick sort being the faster of the two algorithms. In some cases betting out merge sort by almost three times.

Although I was expecting the order in which the algorithms would finish sorting, I was surprised at how slow the three simpler algorithms actually are. The bubble sort algorithm is obviously not a practical choice when it comes to sorting large arrays of data and should be avoided in cases when time is important. The same goes for the insertion algorithms, which don’t take as long as bubble sort, but are not very efficient. Both merge sort and quick sort are surprisingly quick. Although, it’s not too surprising when looking at how to code works. Merge sort splits arrays in half repeatedly and uses recursion. It also requires two separate functions. Quick sort is similar to merge sort but breaks down the array into even smaller pieces, resulting in a faster sort. In most cases when dealing with large arrays of data quick sort should be implemented.

(All single graphs below)