Assignment 6: Sorting Report

Joshua Barrs — Section 1

May 17, 2019

Overall, the time differences were not much different than what I expected, though I was surprised by the run-time of QuickSort. Although all of the sorting algorithms implemented took the same amount of time in terms of seconds, the milliseconds run-time was different.

I was surprised by the amount of time it took QuickSort to sort the list of values. It took it about 0.0509 seconds. However, I was expecting QuickSort to take the least amount of time to sort the values. However, other factors may come into play, such as the way in which I implemented the sorting algorithm.

BubbleSort took about 0.0073 seconds, or 7 milliseconds. Again, I was expecting BubbleSort to take slightly longer than QuickSort.

Insertion Sort came out at about 0.0066 seconds, or around 7 seconds, similar to BubbleSort. This was not surprising, since Insertion Sort and Bubble Sort take $O(n^2)$ time in the average case.

Selection Sort took about 0.0067 seconds, or 7 milliseconds. Again, this was expected, as SelectionSort runs in $O(n^2)$ time.

In picking an algorithm, you must take into account factors such as the size of the data set you would like to sort. Algorithms like QuickSort work best in sorting large amounts of data since they usually run in $O(n^2)$ time. Memory must also come into play. Algorithms such as merge sort require more memory.

Also, regarding programming language, the syntax required for C++ may have been a factor as to why QuickSort took longer than expected. Another possibility is that QuickSort will run in better time overall on large data sets.

Finally, using empirical analysis, I am testing these algorithms on a relatively small amount of data. Using very large data sets would most likely produce different results.