Assignment 6: Sorting Algorithms

Sammie Walker

December 2019

1 Bubble Sort

To perform Bubble Sort: move through array comparing neighboring pairs of elements and swap them if they are not in the correct order. The time complexity is O(n^2). In my program, the run time was 0.006 milliseconds.

2 Insertion Sort

To perform Insertion Sort: use a marker, elements to the left are partially sorted. Check the elements to right of marker and move to correct place on the left. The time complexity is O(n^2). In my program, the run time was 0.011 milliseconds.

3 Selection Sort

To perform Selection Sort: find the smallest element in the array, bring it to the front, and repeat on rest of array. The time complexity is O(n^2). In my program, the run time was 0.004 milliseconds.

4 Quick Sort

To perform Quick Sort: select a pivot point and move all elements bigger than the pivot to the right and smaller to the left. Perform this recursively. Most often the time complexity is O($nlogn$), but with a bad pivot it might be O(n^2). In my program, the run time was 0.002 milliseconds.

5 Analysis

According to my results, Quick Sort is the fastest sorting algorithm, Selection Sort is second fastest, then Bubble Sort, and Insertion Sort is the slowest. I am not too surprised by these results because though Quick Sort is more difficult to implement it is generally considered the fastest sorting algorithm. Although Bubble Sort is one of the simplest algorithms to implement, it's efficiency decreases dramatically as the number of elements increase so I'm not surpised that it is one of the slower algorithms. Typically, Insertion Sort will perform less comparisons than Selection Sort, especially with smaller arrray's so I am a little surprised that Insertion Sort was the slowest.