|  |  |  |
| --- | --- | --- |
| **Selection Sort** | | |
| **List Size** | **Comparisons** | **Time (seconds)** |
| **1,000 (observed)** | 499500 | 0.01350545883178711 |
| **2,000 (observed)** | 1999000 | 0.056432247161865234 |
| **4,000 (observed)** | 7998000 | 0.22605586051940918 |
| **8,000 (observed)** | 31996000 | 0.9038076400756836 |
| **16,000 (observed)** | 127992000 | 3.729015827178955 |
| **32,000 (observed)** | 511984000 | 15.072154760360718 |
| **100,000 (estimated)** | 4999843750 | 359.43771386146545 |
| **500,000 (estimated)** | 124996093750 | 1797.1885693073273 |
| **1,000,000 (estimated)** | 499984375000 | 3594.3771386146545 |
| **10,000,000 (estimated)** | 4.9998438e+13 | 35943.771386146545 |

|  |  |  |
| --- | --- | --- |
| **Insertion Sort** | | |
| **List Size** | **Comparisons** | **Time (seconds)** |
| **1,000 (observed)** | 247986 | 0.011508464813232422 |
| **2,000 (observed)** | 1018717 | 0.05000162124633789 |
| **4,000 (observed)** | 3995264 | 0.1932220458984375 |
| **8,000 (observed)** | 16112194 | 0.7844054698944092 |
| **16,000 (observed)** | 64667449 | 3.185988426208496 |
| **32,000 (observed)** | 257507119 | 13.47880744934082 |
| **100,000 (estimated)** | 2505274077 | 179.3093614578247 |
| **500,000 (estimated)** | 62867948974 | 896.5468072891235 |
| **1,000,000 (estimated)** | 251471795898 | 1793.093614578247 |
| **10,000,000 (estimated)** | 2.514718e+13 | 17930.93614578247 |

1. Which sort do you think is better? Why?

I think insertion sort is better because from the observations when the code was ran it had less comparisons and was faster.

1. Which sort is better when sorting a list that is already sorted (or mostly sorted)? Why?

Insertion sort, because where selection sort that makes compares an item to every other item with insertion sort once the item finds its spot in the sorted list it moves to the next item to be sorted.

1. You probably found that insertion sort had about half as many comparisons as selection sort. Why? Why are the times for insertion sort not half what they are for selection sort? (For part of the answer, think about what insertion sort has to do more of compared to selection sort.)

For insertion sort comparisons, on the best case they make 1 comparison while on the worst case it makes the compares against every element. Averaging the number of comparisons makes it about half the amount as selection sort. For it not taking half the time, it has to do more swaps than in selection sort taking up some time.

|  |  |  |
| --- | --- | --- |
|  | **Number of Quicksort Comparisons** | |
| **Starting List** | pivot = first | pivot = median of 3 |
| Ordered, ascending |  |  |
| n = 100 | 4950 | 4950 |
| n = 200 | 19900 | 19900 |
| n = 400 | 79800 | 79800 |
| n = 800 | 319600 | 319600 |
| Random |  |  |
| n = 100 (average 10 runs) | 650.8 | 627.1 |
| n = 200 (average 10 runs) | 1606.6 | 1514.0 |
| n = 400 (average 10 runs) | 3686.2 | 3538.9 |
| n = 800 (average 10 runs) | 8621.4 | 8229.9 |
|  |  |  |
| Observed Big O() behavior, ordered with pivot = first : nlogn | | |
| Observed Big O() behavior, ordered with pivot = median of 3 : nlogn | | |
| Observed Big O() behavior, random with pivot = first : nlogn | | |
| Observed Big O() behavior, random with pivot = median of 3 : nlogn | | |
| For random list, observation regarding using first vs. median of 3 : using the median was a bit faster but the time was very similar. | | |