|  |  |
| --- | --- |
| Bubble Sort Ascending | time (s) |
| 1000 | 0.15751794999960111 |
| 5000 | 3.9371042499988107 |
| 10000 | 15.90362830000231 |
| 15000 | 35.86455940000451 |

|  |  |
| --- | --- |
| Bubble sort  decending | time (s) |
| 1000 | 0.45081875000323635 |
| 5000 | 11.11926439999661 |
| 10000 | 46.977774949999 |
| 15000 | 100.79714214999694 |

|  |  |
| --- | --- |
| Bubble sort  Nearly Sorted | time (s) |
| 1000 | 0.23386144999676617 |
| 5000 | 4.572605999994266 |
| 10000 | 17.797906850006257 |
| 15000 | 39.71225930000219 |

|  |  |
| --- | --- |
| insertion Sort Ascending | time (s) |
| 1000 | 0.0008862499962560833 |
| 5000 | 0.004070350056281313 |
| 10000 | 0.008534399996278808 |
| 15000 | 0.011491599987493828 |

|  |  |
| --- | --- |
| insertion Sort Decending | time (s) |
| 1000 | 0.4981320000078995 |
| 5000 | 11.737034649995621 |
| 10000 | 47.74308900000324 |
| 15000 | 110.11639055001433 |

|  |  |
| --- | --- |
| insertion Sort Nearly Sorted | time (s) |
| 1000 | 0.04781935000210069 |
| 5000 | 0.7386335500050336 |
| 10000 | 3.0117071499698795 |
| 15000 | 6.532791049998195 |

|  |  |
| --- | --- |
| selection Sort Ascending | time (s) |
| 1000 | 0.15411615004995838 |
| 5000 | 3.8810160499997437 |
| 10000 | 15.74981204996584 |
| 15000 | 35.44430144998478 |

|  |  |
| --- | --- |
| selection Sort Decending | time (s) |
| 1000 | 0.19101795001188293 |
| 5000 | 3.8956193000194617 |
| 10000 | 15.856256899947766 |
| 15000 | 35.156005249999 |

|  |  |
| --- | --- |
| selection Sort nearly sorted | time (s) |
| 1000 | 0.1981620999868028 |
| 5000 | 3.8620200499426574 |
| 10000 | 16.252691000059713 |
| 15000 | 36.45005329995183 |

From the gathered results comparing insertion, bubble and selection sorting algorithms, we can conclude that when the N value increases in each of the O(N^2) sorting algorithms, the longer it takes for the values to be sorted. Insertion sorting is a very stable sorting algorithm that does lots of swaps and through the results, it is the fastest for ascending and nearly sorted data but is the slowest for descending data. Because minimal swaps have to be made, insertion sorting sorts nearly sorted data the fastest. The simple to implement bubble sorting algorithm does constant swapping per pass and is the 2nd fastest for nearly sorted and descending data while being the slowest for ascending data. Bubble sorting requires smaller elements to start not very far from their final sorted position for it to have a fast nearly sorted time that is similar to insertion sort. The unstable but simple to implement selecting sorting algorithm is the fastest for descending data, the 2nd fastest for ascending data and slowest for nearly sorted data.