**NAME**: Edgar Ontieros

**HYPOTHESIS:**

I plan to test the Selection ( O(n2) ), Insertion ( O(n2) ), and Bubble ( O(n2) ) sorting algorithms against the JAVA api built-in Quicksort algorithm ( n log(n) ); I believe the Selection, Insertion, and Bubble sorting algorithms will be more efficient (faster) than the JAVA api built-in Quicksort algorithm on data sets less than 100 elements, and the Quicksort will pick up in efficiency around 1000+ data elements (and consistently/significantly outperform the ( O(n2) ) algorithms); each data set size will be tested 3 times.

**RESULTS:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ARRAY SIZE** | **SelectionSort** | **InsertionSort** | **BubbleSort** | **Built-in QuickSort** |
| **10** | **16014.33333** | **22378.66667** | **17451.33333** | **330345.3333** |
| **100** | **114768.6667** | **65802** | **202026** | **346359.3333** |
| **1000** | **1540755** | **1064126.333** | **3354266.333** | **696927.6667** |
| **10000** | **89214334.33** | **42509506.33** | **220419558.7** | **2564948.667** |
| **100000** | **8897248188** | **3727876192** | **20784216565** | **11486217.33** |

**CONCLUSION:**

For data sets less than 1,000 elements, the Selection, Insertion, and Bubble sorting algorithms ( O(n2) ) were mostly consistent with each other, showing greater efficiency than the Built-in Quicksort ( O(n log(n) ); the fastest of the three O(n2) sorting algorithms for data sets less than 1,000 elements was the Insertion sorting algorithm. For data sets equal to and greater than 1,000 elements, the Built-in QuickSort algorithm showed consistent and significant display of efficiency compared to the three O(n2) sorting algorithms; the Insertion sorting algorithm was still the fastest of the three O(n2) sorting algorithms for data sets greater than 1,000 elements, but the Built-in QuickSort algorithm exceled by a large margin.