**Test Results:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sort** | **Run #1** | **Run #2** | **Run #3** | **Run #4** | **Run #5** | **Average** |
| Bubble Sort | 28,810,659 | 28,575,543 | 29,019,372 | 28,875,685 | 28,976,259 | *28,851,504* |
| Insertion Sort | 11,150,047 | 11,112,954 | 11,299,969 | 11,171,317 | 11,337,839 | *11,214,425* |
| Merge Sort | 13,479 | 11,704 | 12,174 | 11,588 | 14,566 | *12,702* |
| Quick Sort | 16,512 | 19,679 | 14,926 | 14,683 | 15,111 | *16,182* |

As the test results surmise, the most efficient sort on average was the Merge Sort. Considering merge sort and quick sort both have a Big-O best case of O(n log(n)), they should run at relatively the same time and although quick sort is more efficient in its use of memory, these factors can change based on machine, the numbers and a myriad of other variables. However, our O(n^2) algorithms of bubble sort and insertion sort pictures show identical pictures of what to expect using these sorts on large arrays with bubble sort being the slowest and the insertion sort being only somewhat more efficient.