# Project 2 Analysis
Run Times of Sorting Algorithms

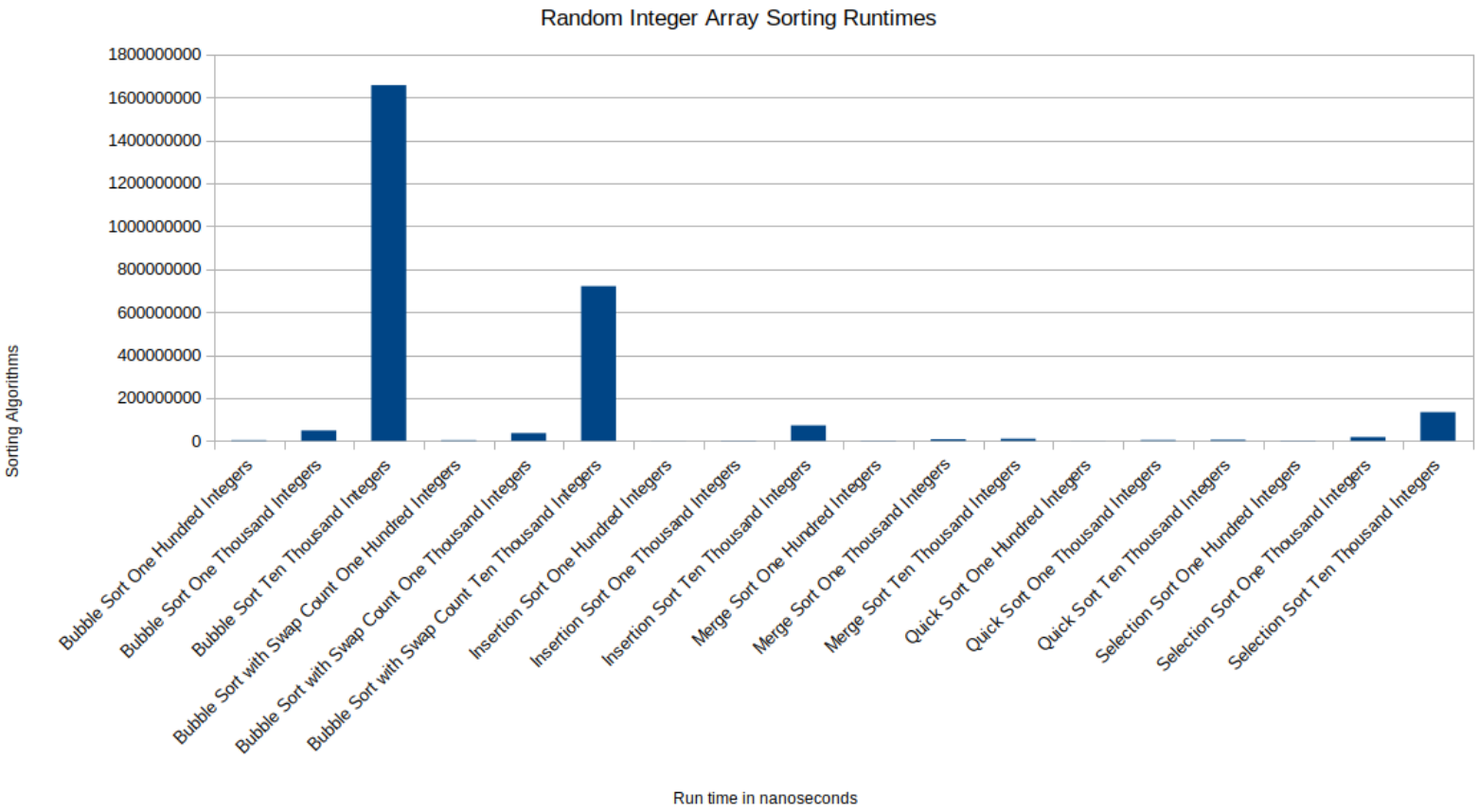
Phillip Janowski

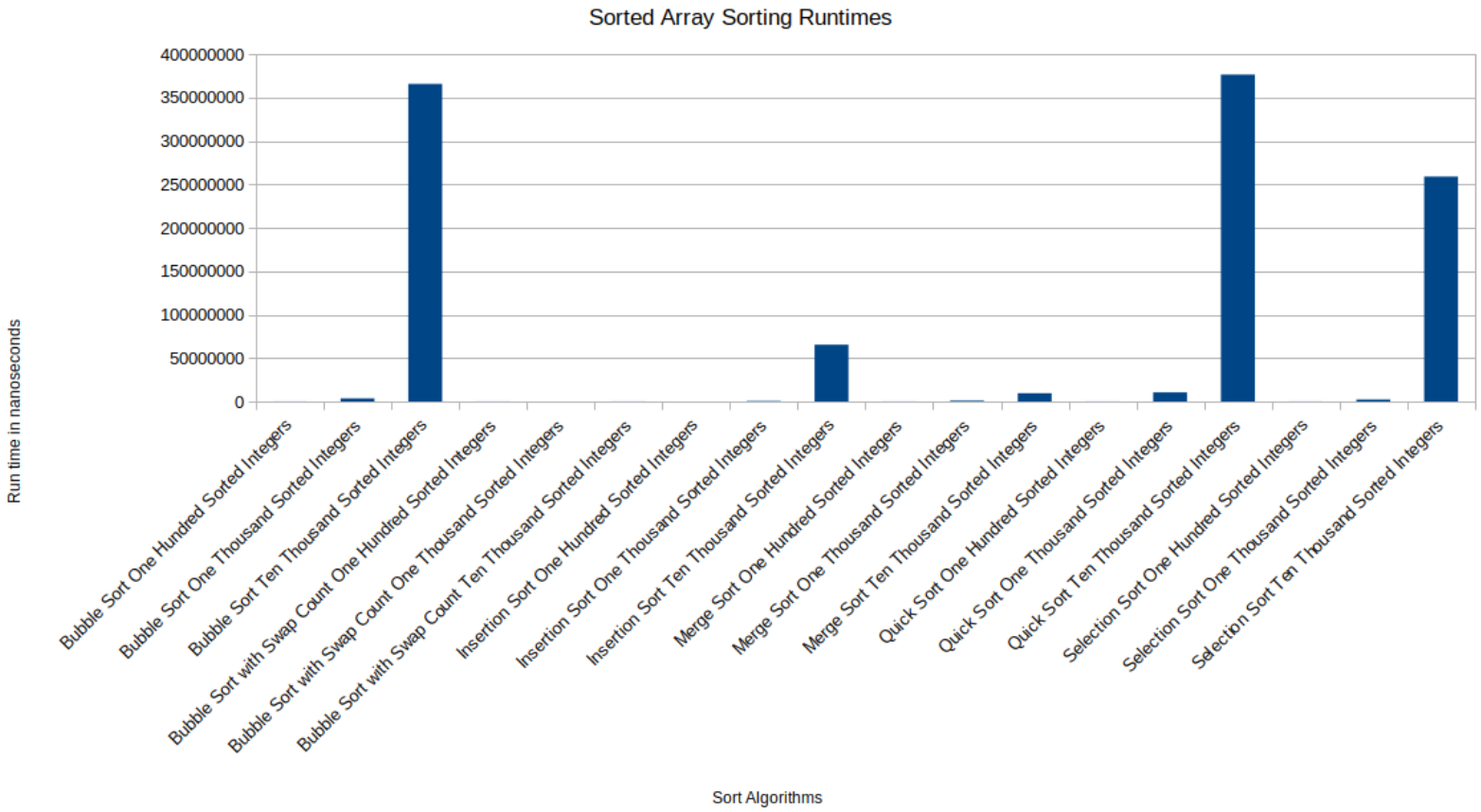
October 22, 2018

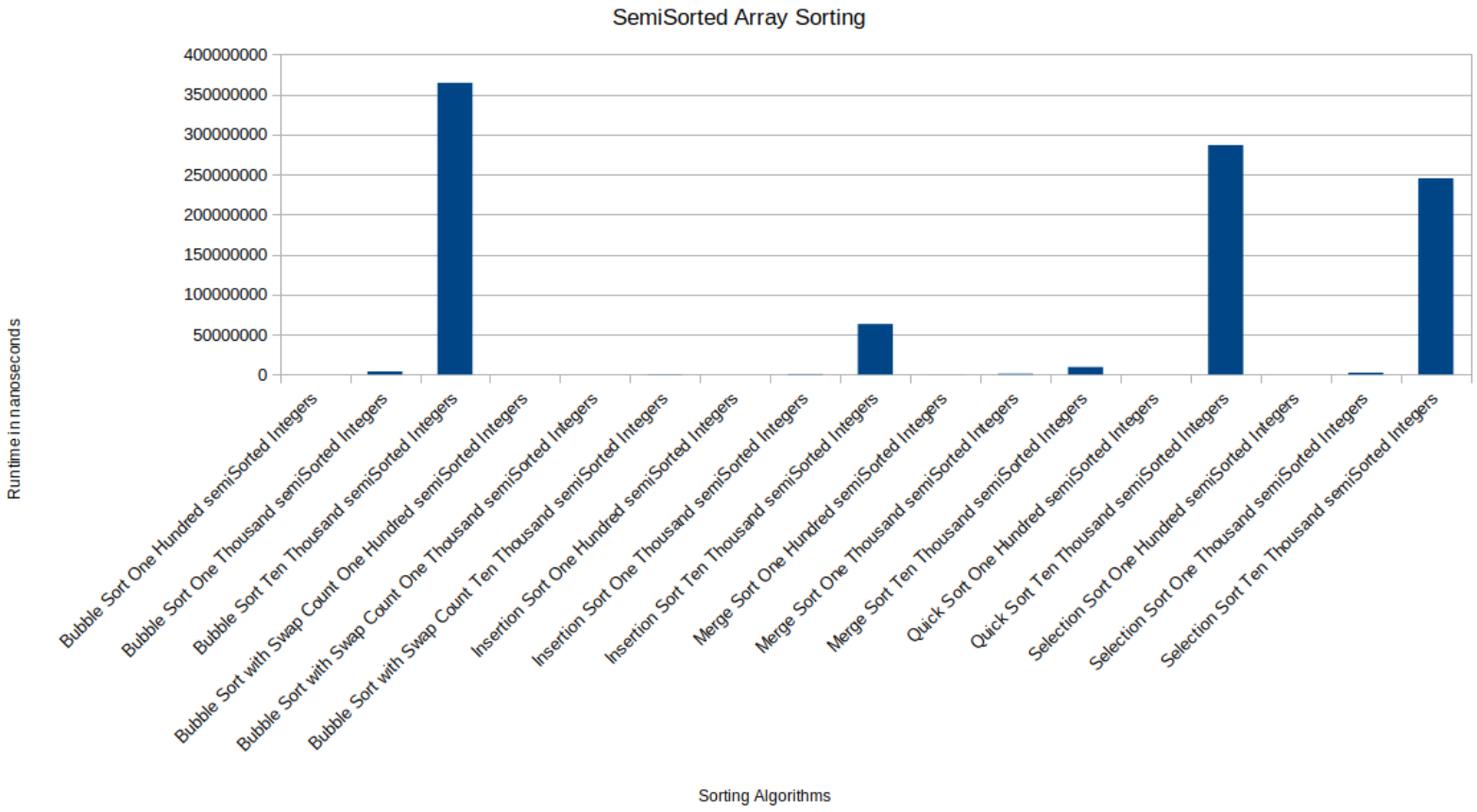
## 1 Theoretical Time Complexities

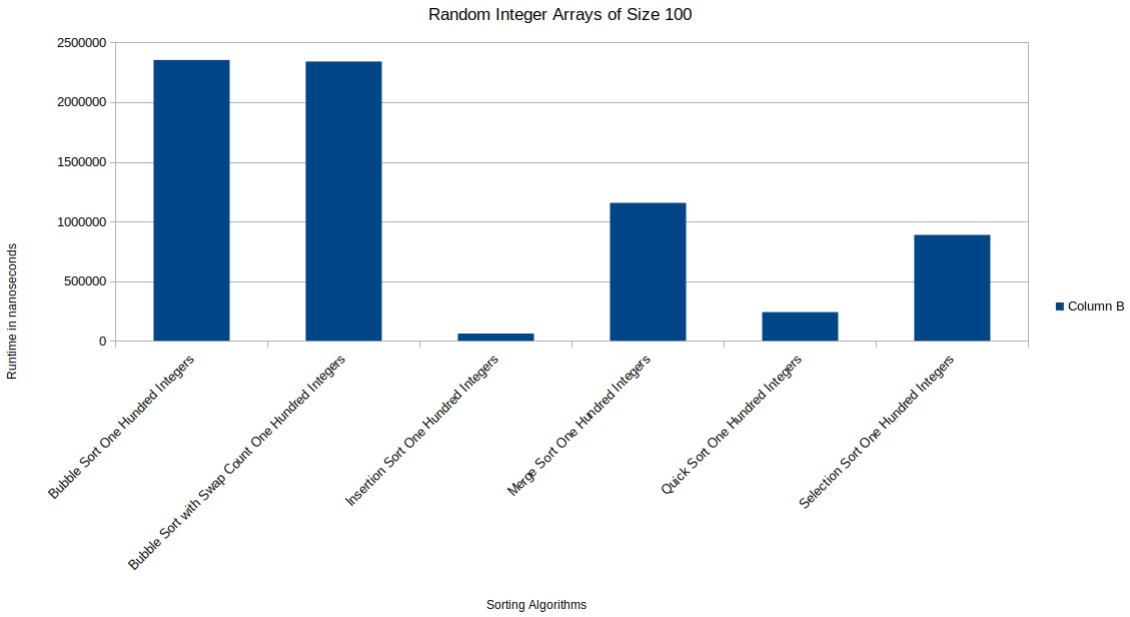
| Algorithm | Best Case | Average Case | Worst Case |
|---|---|---|---|
| Quick Sort | $O(n\log(n))$ | $O(n\log(n))$ | $O(n^2)$ |
| Merge Sort | $O(n\log(n))$ | $O(n\log(n))$ | $O(n\log(n))$ |
| Bubble Sort | $O(n)$ | $O(n^2)$ | $O(n^2)$ |
| Bubble Sort With Swap Counting | $O(1)$ | $O(n\log(n))$ | $O(n^2)$ |
| Insertion Sort | $O(n)$ | $O(n^2)$ | $O(n^2)$ |
| Selection Sort | $O(n^2)$ | $O(n^2)$ | $O(n^2)$ |

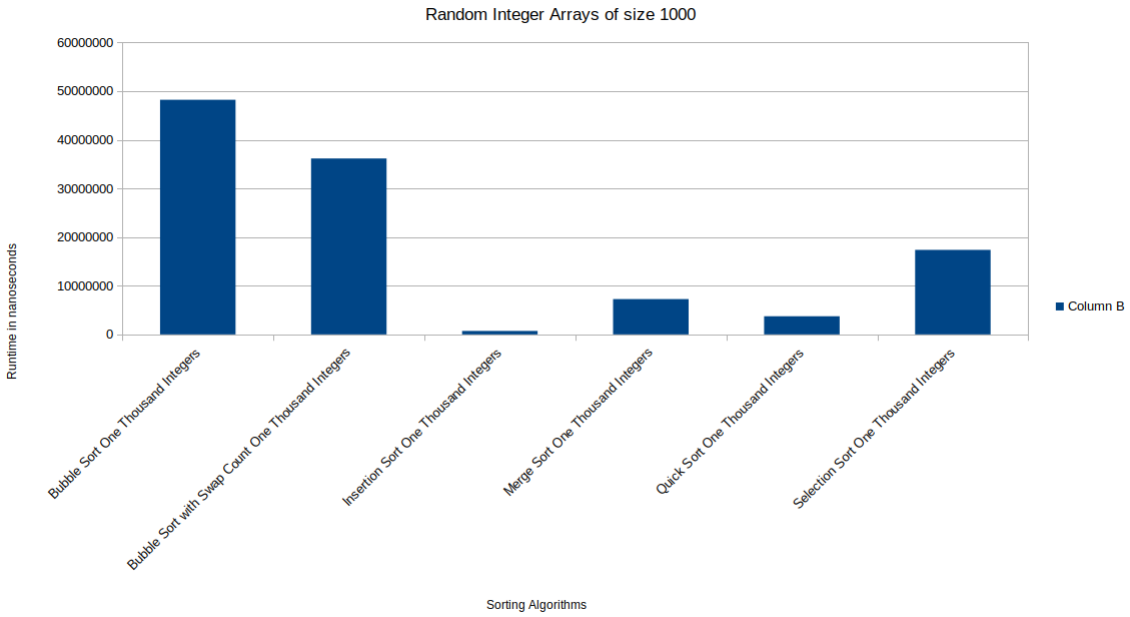
## 2 Observed Times

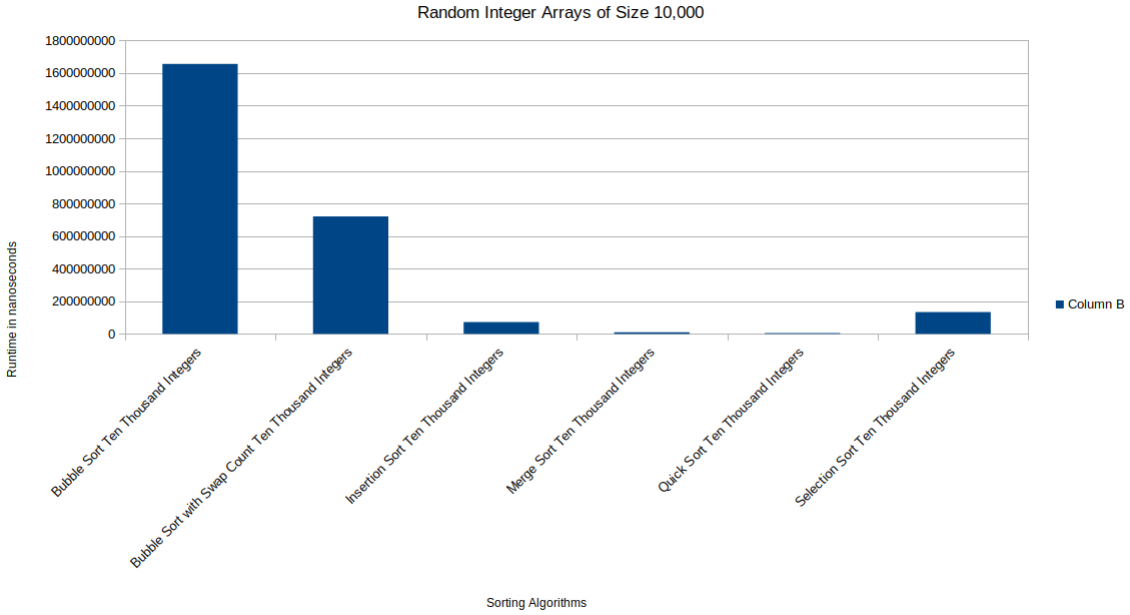


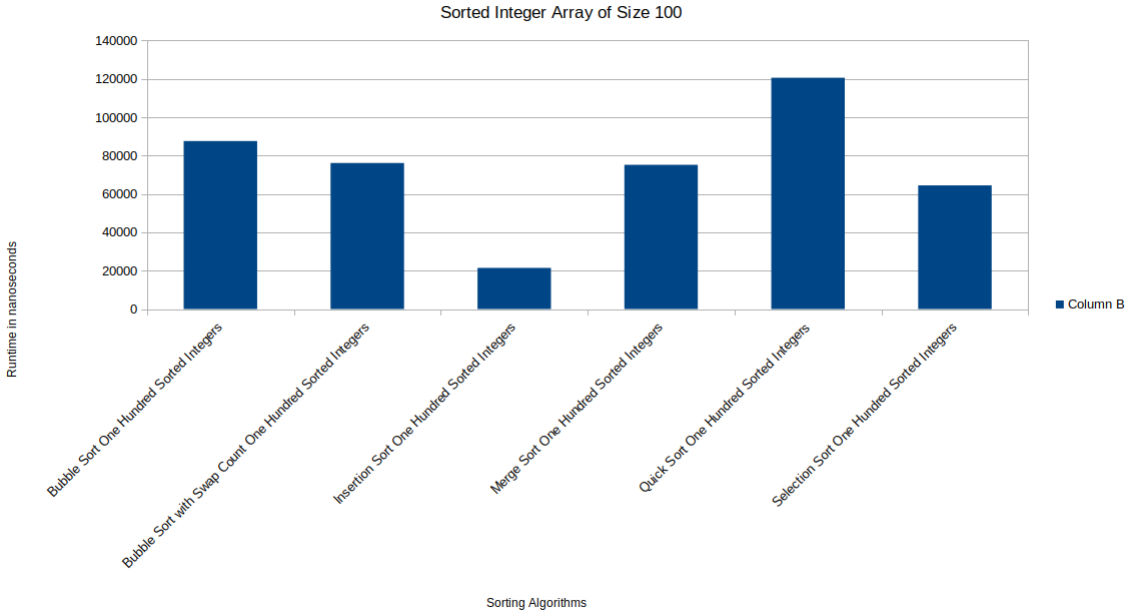

Sorted Array Sorting Runtimes
Run time in nanoseconds
400000000
350000000
300000000
250000000
200000000
150000000
100000000
50000000
0
Bubble Sort One Hundred Sorted Integers
Bubble Sort One Thousand Sorted Integers
Bubble Sort Ten Thousand Sorted Integers
Bubble Sort with Swap Count One Hundred Sorted Integers
Bubble Sort with Swap Count One Thousand Sorted Integers
Bubble Sort with Swap Count Ten Thousand Sorted Integers
Insertion Sort One Hundred Sorted Integers
Insertion Sort One Thousand Sorted Integers
Insertion Sort Ten Thousand Sorted Integers
Merge Sort One Hundred Sorted Integers
Merge Sort One Thousand Sorted Integers
Merge Sort Ten Thousand Sorted Integers
Quick Sort One Hundred Sorted Integers
Quick Sort One Thousand Sorted Integers
Quick Sort Ten Thousand Sorted Integers
Selection Sort One Hundred Sorted Integers
Selection Sort One Thousand Sorted Integers
Selection Sort Ten Thousand Sorted Integers
Sort Algorithms

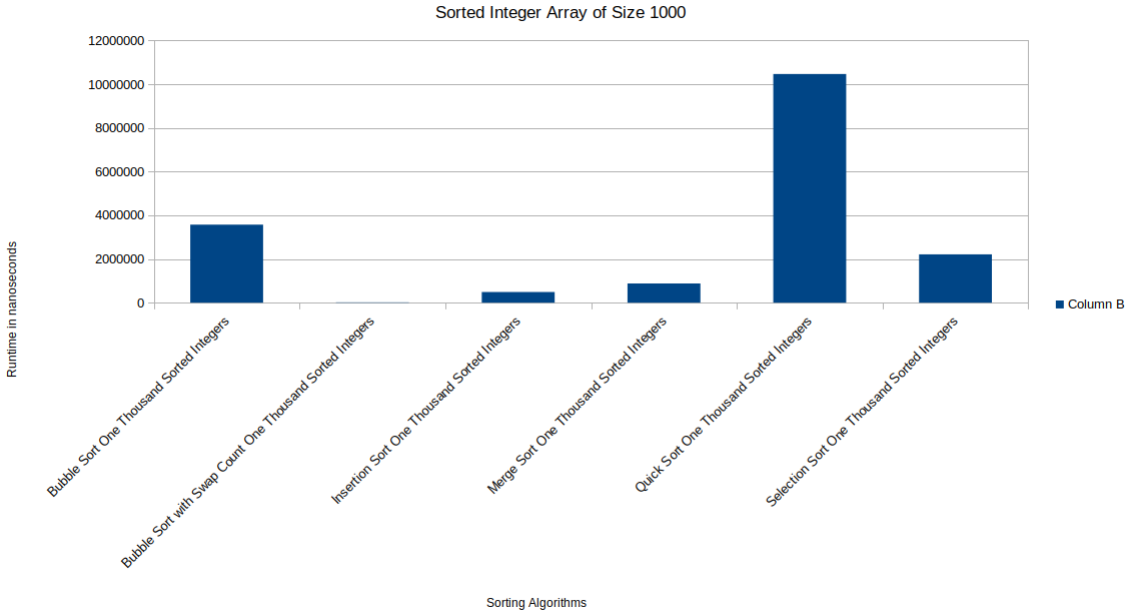

SemiSorted Array Sorting
400000000
350000000
300000000
250000000
200000000
150000000
100000000
50000000
0
Runtime in nanoseconds
Bubble Sort One Hundred semiSorted Integers
Bubble Sort One Thousand semiSorted Integers
Bubble Sort Ten Thousand semiSorted Integers
Bubble Sort with Swap Count One Hundred semiSorted Integers
Bubble Sort with Swap Count One Thousand semiSorted Integers
Bubble Sort with Swap Count Ten Thousand semiSorted Integers
Insertion Sort One Hundred semiSorted Integers
Insertion Sort One Thousand semiSorted Integers
Insertion Sort Ten Thousand semiSorted Integers
Merge Sort One Hundred semiSorted Integers
Merge Sort One Thousand semiSorted Integers
Merge Sort Ten Thousand semiSorted Integers
Quick Sort One Hundred semiSorted Integers
Quick Sort Ten Thousand semiSorted Integers
Selection Sort One Hundred semiSorted Integers
Selection Sort One Thousand semiSorted Integers
Selection Sort Ten Thousand semiSorted Integers
Sorting Algorithms

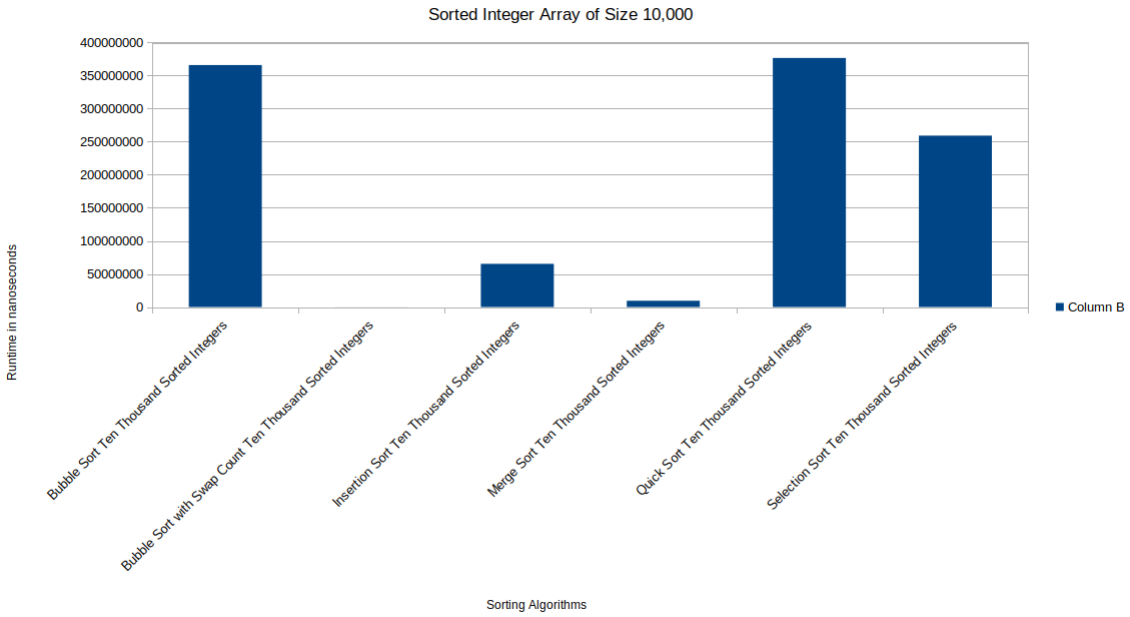

Random Integer Arrays of Size 100
2500000
2000000
1500000
1000000
500000
0
Runtime in nanoseconds
Bubble Sort One Hundred Integers
Bubble Sort with Swap Count One Hundred Integers
Insertion Sort One Hundred Integers
Merge Sort One Hundred Integers
Quick Sort One Hundred Integers
Selection Sort One Hundred Integers
Column B
Sorting Algorithms

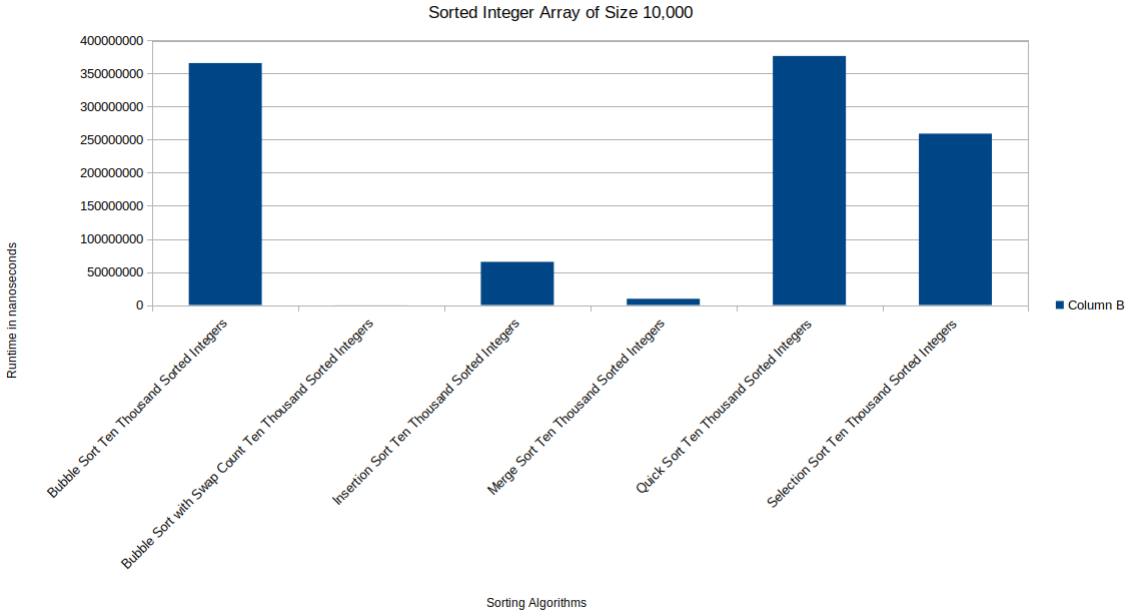

Random Integer Arrays of size 1000
60000000
50000000
40000000
30000000
20000000
10000000
0
Runtime in nanoseconds
Bubble Sort One Thousand Integers
Bubble Sort with Swap Count One Thousand Integers
Insertion Sort One Thousand Integers
Merge Sort One Thousand Integers
Quick Sort One Thousand Integers
Selection Sort One Thousand Integers
Column B
Sorting Algorithms

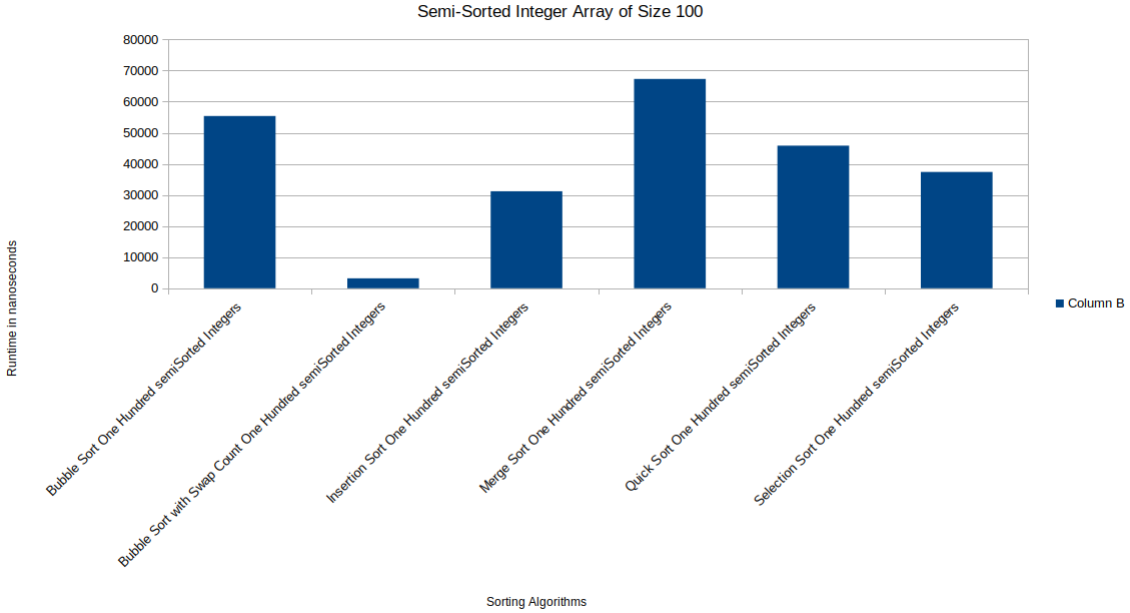

Random Integer Arrays of Size 10,000
1800000000
1600000000
1400000000
1200000000
1000000000
800000000
600000000
400000000
200000000
0
Runtime in nanoseconds
Bubble Sort Ten Thousand Integers
Bubble Sort with Swap Count Ten Thousand Integers
Insertion Sort Ten Thousand Integers
Merge Sort Ten Thousand Integers
Quick Sort Ten Thousand Integers
Selection Sort Ten Thousand Integers
Column B
Sorting Algorithms

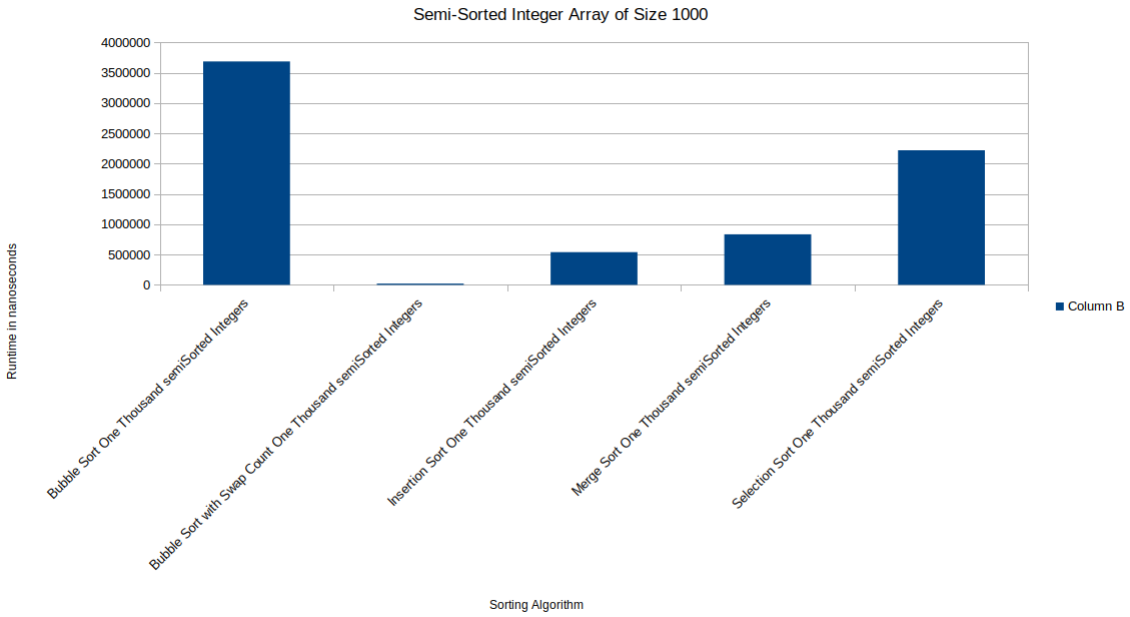

Sorted Integer Array of Size 100
140000
120000
100000
80000
60000
40000
20000
0
Runtime in nanoseconds
Bubble Sort One Hundred Sorted Integers
Bubble Sort with Swap Count One Hundred Sorted Integers
Insertion Sort One Hundred Sorted Integers
Merge Sort One Hundred Sorted Integers
Quick Sort One Hundred Sorted Integers
Selection Sort One Hundred Sorted Integers
Column B
Sorting Algorithms

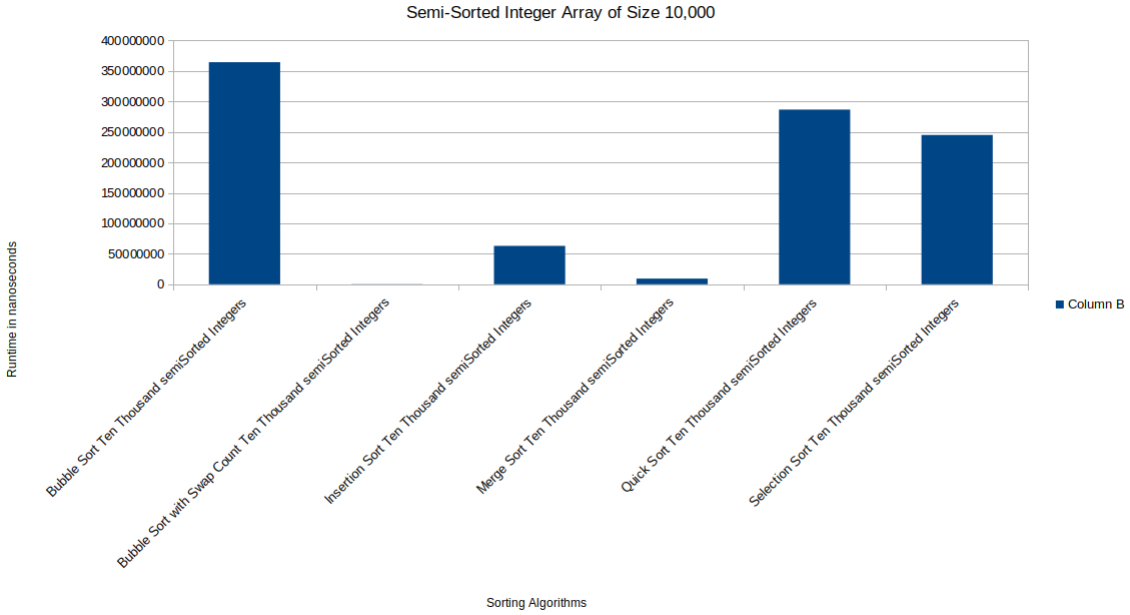

Sorted Integer Array of Size 1000
12000000
10000000
8000000
6000000
4000000
2000000
0
Runtime in nanoseconds
Bubble Sort One Thousand Sorted Integers
Bubble Sort with Swap Count One Thousand Sorted Integers
Insertion Sort One Thousand Sorted Integers
Merge Sort One Thousand Sorted Integers
Quick Sort One Thousand Sorted Integers
Selection Sort One Thousand Sorted Integers
Column B
Sorting Algorithms


Sorted Integer Array of Size 10,000
400000000
350000000
300000000
250000000
200000000
150000000
100000000
50000000
0
Runtime in nanoseconds
Bubble Sort Ten Thousand Sorted Integers
Bubble Sort with Swap Count Ten Thousand Sorted Integers
Insertion Sort Ten Thousand Sorted Integers
Merge Sort Ten Thousand Sorted Integers
Quick Sort Ten Thousand Sorted Integers
Selection Sort Ten Thousand Sorted Integers
Column B
Sorting Algorithms


Sorted Integer Array of Size 10,000
Runtime in nanoseconds
400000000
350000000
300000000
250000000
200000000
150000000
100000000
50000000
0
Bubble Sort Ten Thousand Sorted Integers
Bubble Sort with Swap Count Ten Thousand Sorted Integers
Insertion Sort Ten Thousand Sorted Integers
Merge Sort Ten Thousand Sorted Integers
Quick Sort Ten Thousand Sorted Integers
Selection Sort Ten Thousand Sorted Integers
Column B
Sorting Algorithms


Semi-Sorted Integer Array of Size 100
Runtime in nanoseconds
80000
70000
60000
50000
40000
30000
20000
10000
0
Bubble Sort One Hundred semiSorted Integers
Bubble Sort with Swap Count One Hundred semiSorted Integers
Insertion Sort One Hundred semiSorted Integers
Merge Sort One Hundred semiSorted Integers
Quick Sort One Hundred semiSorted Integers
Selection Sort One Hundred semiSorted Integers
Column B
Sorting Algorithms


Semi-Sorted Integer Array of Size 1000
Runtime in nanoseconds
4000000
3500000
3000000
2500000
2000000
1500000
1000000
500000
0
Bubble Sort One Thousand semiSorted Integers
Bubble Sort with Swap Count One Thousand semiSorted Integers
Insertion Sort One Thousand semiSorted Integers
Merge Sort One Thousand semiSorted Integers
Selection Sort One Thousand semiSorted Integers
Column B
Sorting Algorithm


Semi-Sorted Integer Array of Size 10,000
Runtime in nanoseconds
400000000
350000000
300000000
250000000
200000000
150000000
100000000
50000000
0
Bubble Sort Ten Thousand semiSorted Integers
Bubble Sort with Swap Count Ten Thousand semiSorted Integers
Insertion Sort Ten Thousand semiSorted Integers
Merge Sort Ten Thousand semiSorted Integers
Quick Sort Ten Thousand semiSorted Integers
Selection Sort Ten Thousand semiSorted Integers
Column B
Sorting Algorithms

# 3 Raw Data

| | |
|---|---|
| Bubble Sort One Hundred Integers | 2340277 |
| Bubble Sort One Hundred Sorted Integers | 87578 |
| Bubble Sort One Hundred semiSorted Integers | 55402 |
| Bubble Sort One Thousand Integers | 36158542 |
| Bubble Sort One Thousand Sorted Integers | 3570585 |
| Bubble Sort One Thousand semiSorted Integers | 3687019 |
| Bubble Sort Ten Thousand Integers | 1655832010 |
| Bubble Sort Ten Thousand Sorted Integers | 365890163 |
| Bubble Sort Ten Thousand semiSorted Integers | 364328717 |
| Bubble Sort with Swap Count One Hundred Integers | 2352939 |
| Bubble Sort with Swap Count One Hundred Sorted Integers | 76115 |
| Bubble Sort with Swap Count One Hundred semiSorted Integers | 3196 |
| Bubble Sort with Swap Count One Thousand Integers | 48218986 |
| Bubble Sort with Swap Count One Thousand Sorted Integers | 14510 |
| Bubble Sort with Swap Count One Thousand semiSorted Integers | 20141 |
| Bubble Sort with Swap Count Ten Thousand Integers | 720552569 |
| Bubble Sort with Swap Count Ten Thousand Sorted Integers | 159855 |
| Bubble Sort with Swap Count Ten Thousand semiSorted Integers | 312103 |

| | |
|---|---|
| Insertion Sort One Hundred Integers | 60212 |
| Insertion Sort One Hundred Sorted Integers | 21443 |
| Insertion Sort One Hundred semiSorted Integers | 31183 |
| Insertion Sort One Thousand Integers | 712859 |
| Insertion Sort One Thousand Sorted Integers | 487260 |
| Insertion Sort One Thousand semiSorted Integers | 539467 |
| Insertion Sort Ten Thousand Integers | 72014508 |
| Insertion Sort Ten Thousand Sorted Integers | 65320081 |
| Insertion Sort Ten Thousand semiSorted Integers | 63007301 |
| Merge Sort One Hundred Integers | 1155576 |
| Merge Sort One Hundred Sorted Integers | 75172 |
| Merge Sort One Hundred semiSorted Integers | 67296 |
| Merge Sort One Thousand Integers | 7258064 |
| Merge Sort One Thousand Sorted Integers | 878313 |
| Merge Sort One Thousand semiSorted Integers | 833963 |
| Merge Sort Ten Thousand Integers | 9450255 |
| Merge Sort Ten Thousand Sorted Integers | 9453833 |
| Merge Sort Ten Thousand semiSorted Integers | 9287946 |
| Quick Sort One Hundred Integers | 239716 |
| Quick Sort One Hundred Sorted Integers | 120574 |
| Quick Sort One Hundred semiSorted Integers | 45863 |
| Quick Sort One Thousand Integers | 3731406 |
| Quick Sort One Thousand Sorted Integers | 10464935 |
| Quick Sort One Thousand semiSorted Integers | 3945559 |
| Quick Sort Ten Thousand Integers | 4627244 |
| Quick Sort Ten Thousand Sorted Integers | 376676526 |
| Quick Sort Ten Thousand semiSorted Integers | 286523626 |
| Selection Sort One Hundred Integers | 886852 |
| Selection Sort One Hundred Sorted Integers | 64451 |
| Selection Sort One Hundred semiSorted Integers | 37396 |
| Selection Sort One Thousand Integers | 17377600 |
| Selection Sort One Thousand Sorted Integers | 2213129 |
| Selection Sort One Thousand semiSorted Integers | 2221025 |
| Selection Sort Ten Thousand Integers | 133593636 |
| Selection Sort Ten Thousand Sorted Integers | 259173774 |
| Selection Sort Ten Thousand semiSorted Integers | 244965256 |

# 4 Analysis

The reduction in running times between the completely random arrays and the sorted and semi-sorted arrays supports the theoretical time complexities to an extent. Bubble sort with swap counts definitely stood out working better than every other sorting algorithms in semi and fully sorted arrays supporting the best case run time of $O(1)$ and $O(n\log(n))$ for best and average, respectively, while the non swap counting version of bubble sort is consistently the worst in these situations having a run time of $O(n^2)$ for average and worst.

We can also see merge sort being classified as slow to middling in smaller integer array sizes, but as array sizes get larger, we see merge sort becoming more efficient compared to that of non partitioning algorithms, supporting its across the board $O(n\log(n))$ time complexity. Quick Sort, instead of getting more efficient as the arrays get larger, gets more efficient as arrays are closer to unsorted compared to other sorting algorithms, supporting that its best case and average cases have the same time complexities of $O(n\log(n))$.

Bubble sort and insertion sort are both relatively consistent in time complexity rankings and growth as their theoretical time complexities support because both have $O(n)$ growth for their best case and $O(n^2)$ for their worst and average. Selection sort is very consistent in its ranking when "sorting" arrays larger than 1000 elements and mostly consistent in its growth, supporting its theoretical time complexity of $O(n^2)$ across the board.

The theoretical time complexities are supported by the experimental data in every case tested.