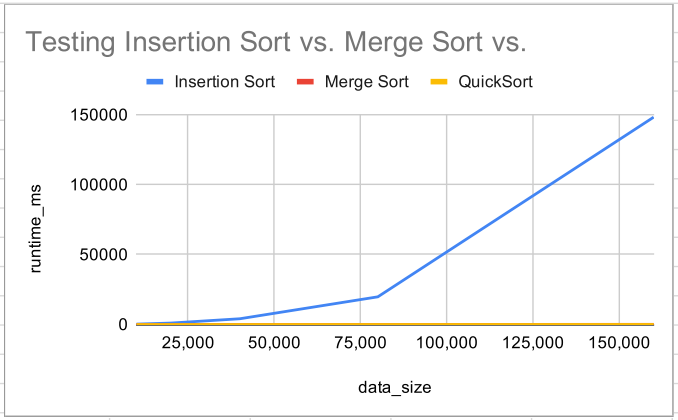
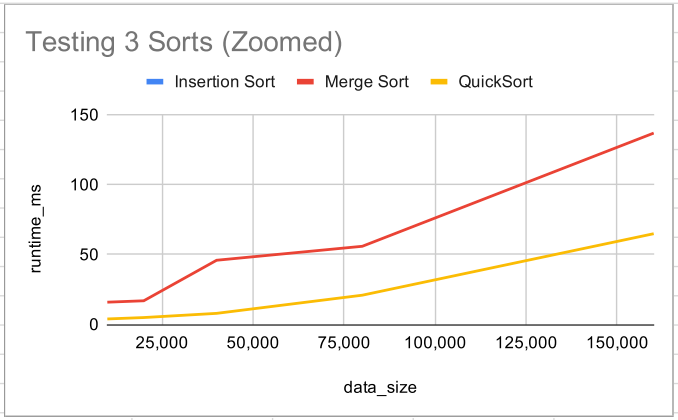
| data_size | Insertion Sort | Merge Sort | QuickSort |
|---|---|---|---|
| 10,000 | 304 | 16 | 4 |
| 20,000 | 1059 | 17 | 5 |
| 40,000 | 4147 | 46 | 8 |
| 80,000 | 19833 | 56 | 21 |
| 160,000 | 148393 | 137 | 65 |

These empiral tests reveal Insertion Sort's average case time complexity of O(n^2), as well as Merge Sort and Quicksort's average case time complexities of O(nlogn), which perform dramatically faster. Additionally, it seems that though both Merge Sort and QuickSort have fall in the same asymptotic class time complexity, QuickSort has smaller hidden constants, giving it shorter runtimes.

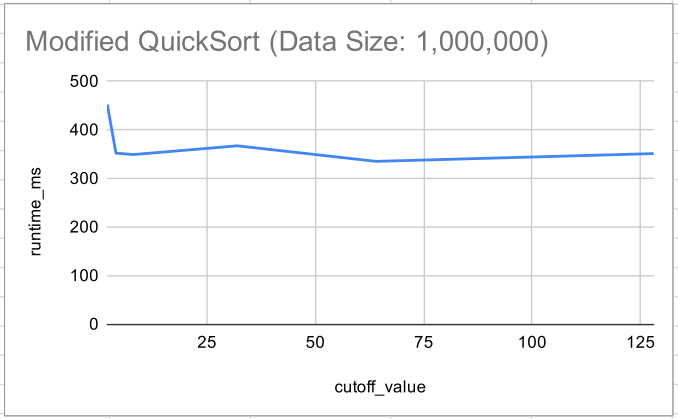



modified quicksort, data size 1,000,000

| cutoff_value | runtime_ms |
|---|---|
| 2 | 452 |
| 4 | 352 |
| 8 | 349 |
| 16 | 355 |
| 32 | 367 |
| 64 | 335 |
| 128 | 351 |

The cutoff of 64 gave the best performance, and the other results indeed show that cutoffs that are too small or too large are not optimal for runtime




cutoff value: 64 (from previous problem)

| data_size | QuickSort | Modified QuickSort |
|---|---|---|
| 100,000 | 48 | 41 |
| 200,000 | 70 | 53 |
| 300,000 | 78 | 102 |
| 400,000 | 113 | 109 |
| 500,000 | 143 | 144 |
| 600,000 | 180 | 182 |

Using the cutoff size of 64, QuickSort and Modified Quicksort have very similar runtimes, with slight alternating optimality in smaller data sizes and increasingly similar runtime in larger dataset sizes