Of the 9 given algorithms only 8 can run on the provided dataset as Dutch flag sort can only sort 0’s,1’s and 2,s which none of the datasets are comprised solely out of. What I found was that Median3Sort was consistently the fastest and varied less than others. Many sorting algorithms where more affected by run to run variance, namely shell sort which does not seem to run too consistently run on run.

Many of the algorithms didn’t run well on the small dataset and improved their time from around 600000ns to 300000ns across the board however this can be down to run averages on the first run. The cutoff time used was 3 seconds so therefore some of the slower sorting algorithms flatline on the final 2 files.

Below are the average times for 10 runs of dutch and bad data files.

|  |  |  |
| --- | --- | --- |
| **Algorithm** | **bad.txt** | **dutch.txt** |
| QuickSortInsertion | 2314860 | 533894910 |
| QuickSort | 1994840 | 537286860 |
| BottomUpMergeSort | 2187540 | 157693390 |
| Median3Sort | 1435120 | 412410910 |
| SelectionSort | 106845860 | 3000000000 |
| InsertionSort | 45394560 | 3000000000 |
| MergeSort | 2375840 | 227606150 |
| ShellSort | 2684730 | 117977380 |

Both Selection and Insertion sort failed to run on the dutch.txt due to timing out however completed bad.txt performing quite poorly, median3sort as before worked the best in bad.txt however was quite behind shell sort in dutch.txt meaning that the dataset of bad suited it far better as it does not care for similarly typed data. Its big o notation must have a bad worst case which was hit in the case of dutch.txt file.