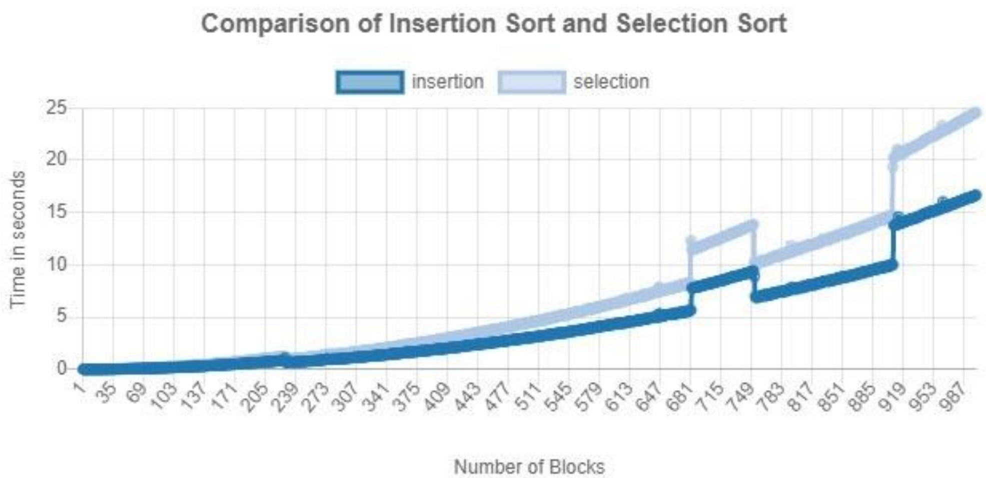
NAME:	Shubham Solanki
UID:	2022301015
SUBJECT	Design and Analysis of Algorithms
AIM:	Graphs of all Sorting Algorithms.
Algorithm:	



Observation

For insertion sort

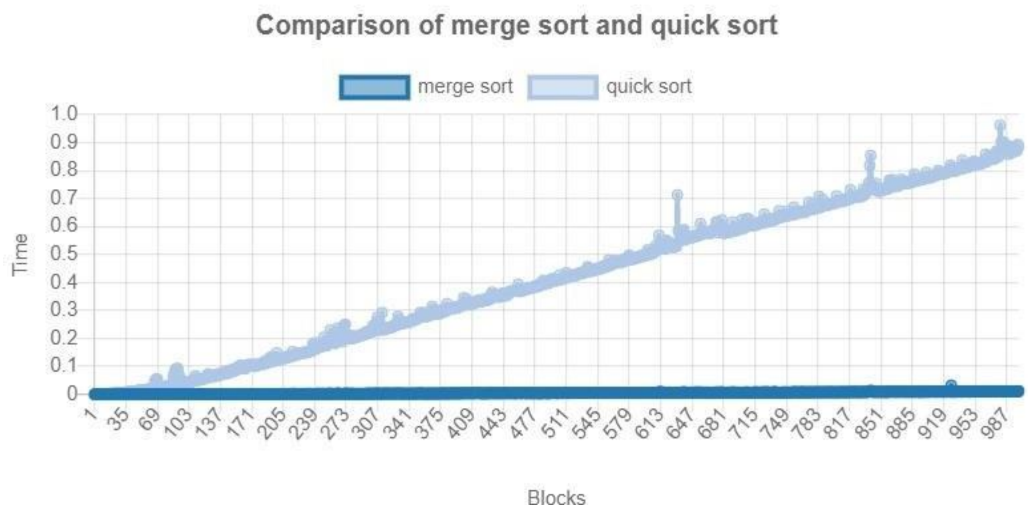
- The insertion sort running time graph shows that the running time increases steadily and consistently as the size of the input data increases.
- The graph shows a recognisable increasing trend, with the slope of the curve steepening as the input data size grows.
- This demonstrates the quadratic nature of insertion sort's running time.

For selection sort

- The graph of the running time of selection sort shows that, like insertion sort, it increases with increasing input data size.

- The rate of rise is still significantly slower than insertion sort, despite the graph's less extreme slope.

This demonstrates that selection sort is more efficient for small input sizes



Observation

For quick sort

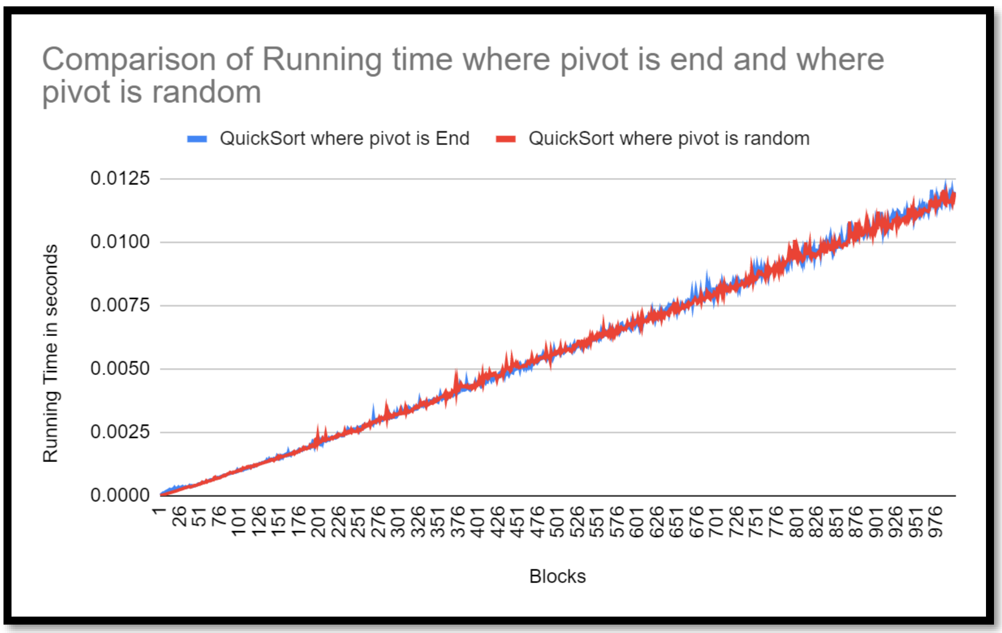
- The graph of the running time of quick sort reveals that as the size of the input data increases, the running time also increases at a steady and consistent rate.
- The graph exhibits a characteristic upward trend, with the slope of the curve becoming steeper as the size of the input data increases.

For merge sort

- The graph of the running time of merge sort shows that it also increases as the size of the input data increases, similar to insertion sort.
- However, the rate of increase is very very slow and very very minute increase in time takes place which is very slow

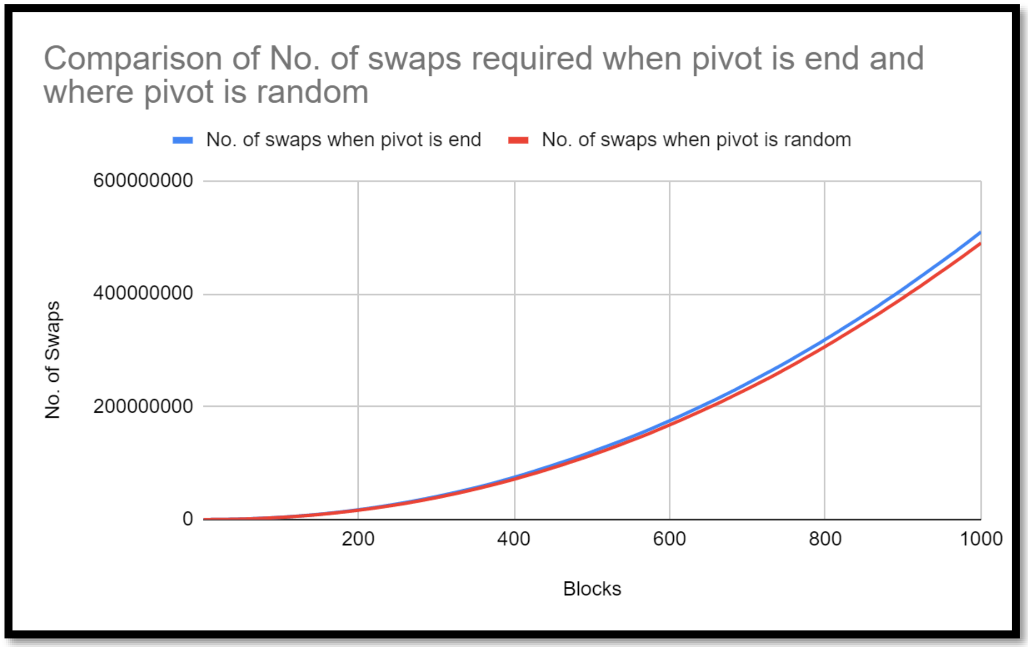
	compared to merge sort, which is evident from the less steep slope of the graph. • This suggests that merge sort is more efficient.

Running time comparison for Different Pivot Positions



Observation

- Here, we can see that the time complexity of quick sort is nearly the same even when varied pivot points are taken into account.
- We can see that a quick sort when the pivot is at a random position takes longer than when the pivot is in an end position at the conclusion of execution.
- Despite the fact that both executions are finished in 0.1 seconds.

	Number of swaps considering different pivot positions  **Observation** • We can see from this that fewer swaps are needed for a quick sort when the pivot is in a random position as opposed to when it is at the end. • The average number of swaps is over 500,000,000. • Throughout the entire execution, the number of swaps keeps rising.
Conclusion:	Thus, we have observed and analysed the graphs of various sorting algorithms