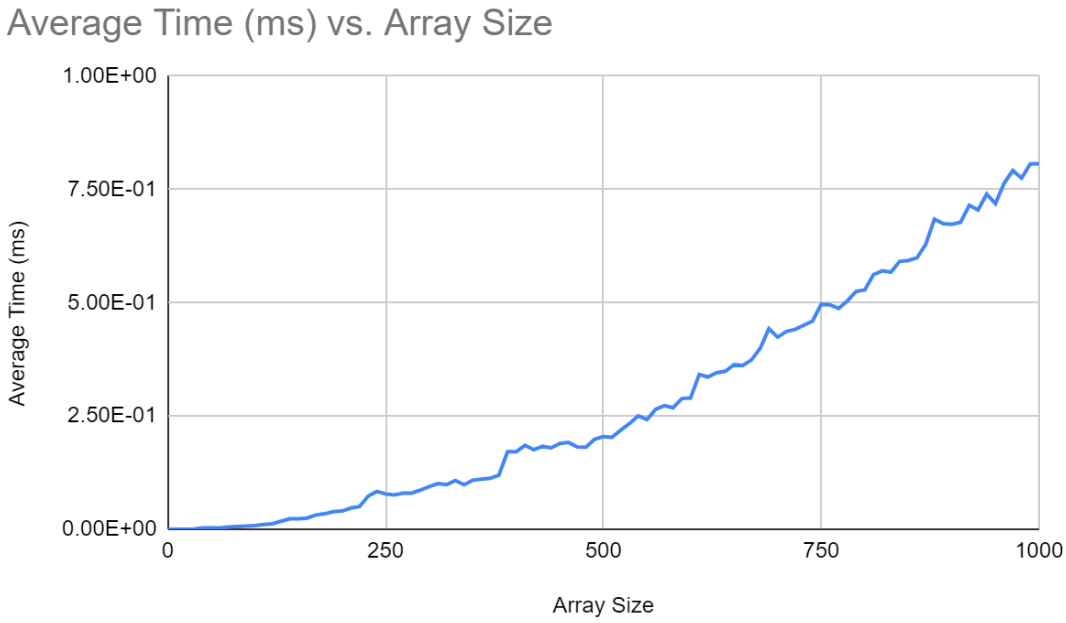
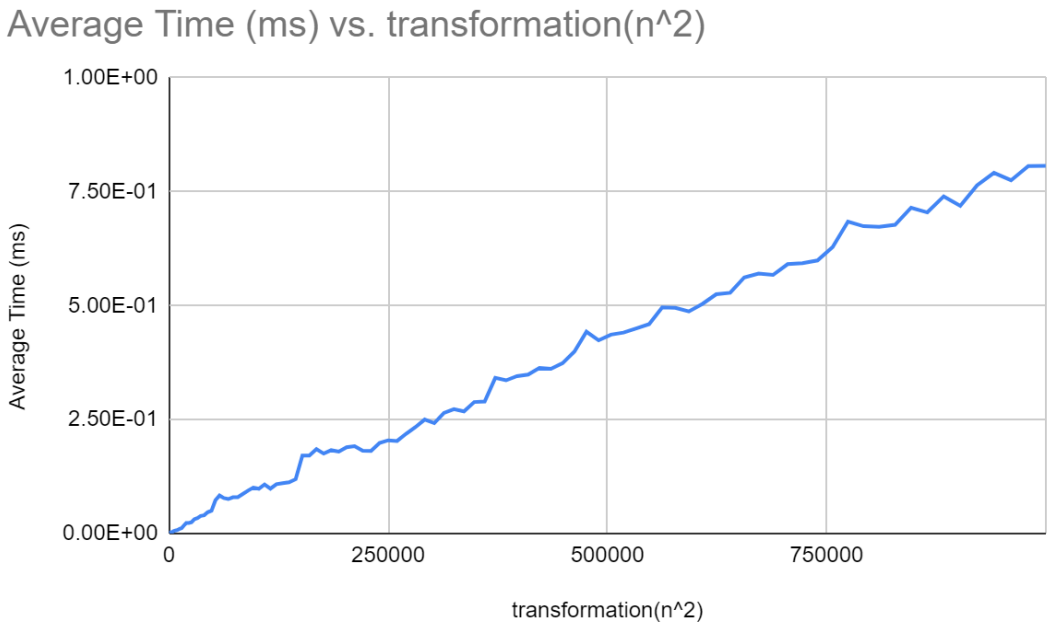
Results for Insertions Sort:

Time is in milliseconds.



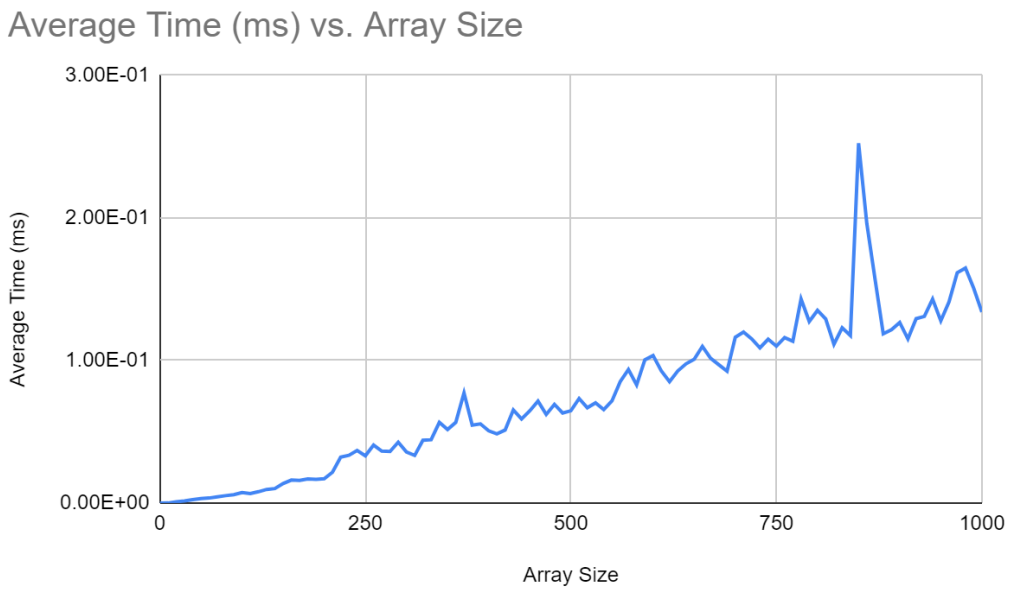
From this plot, we can see that selection sort has worse than linear time. This is because the data points curve upward, growing faster than linear growth.



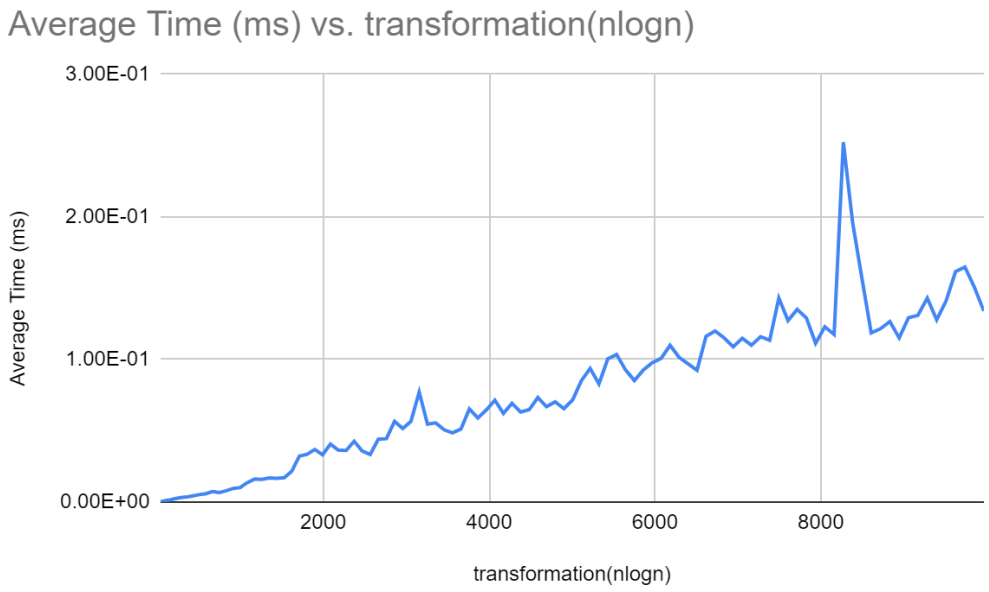
From this plot, we can see that the selection sort has an average run time that is O(n^2) since when we plot the time against the size (n) squared, we observe a linear relationship.

Results for Merge Sort:

Time is in milliseconds.



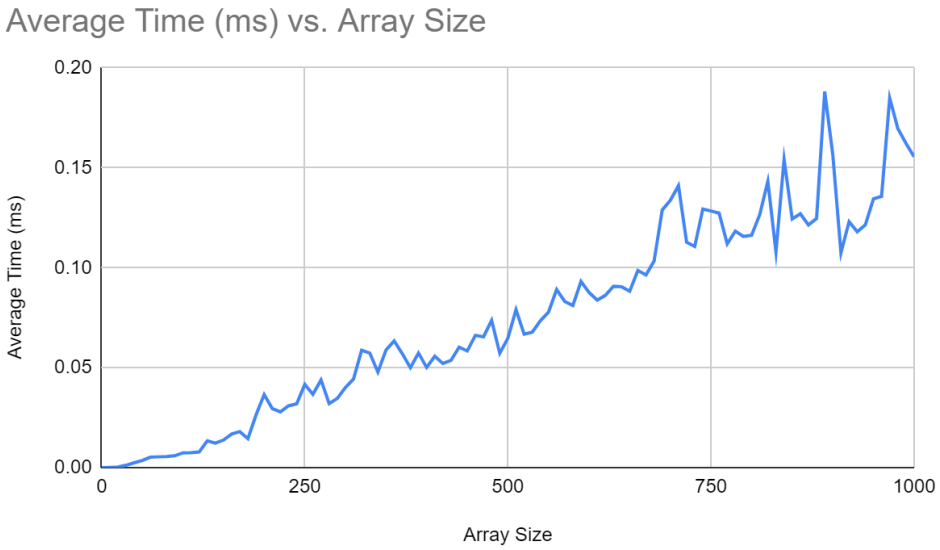
From this plot, we can kind of deduce that it is either linear or better time complexity. But since we know Merge sort has the worst case time complexity of nlog(n) we will do the transformation.



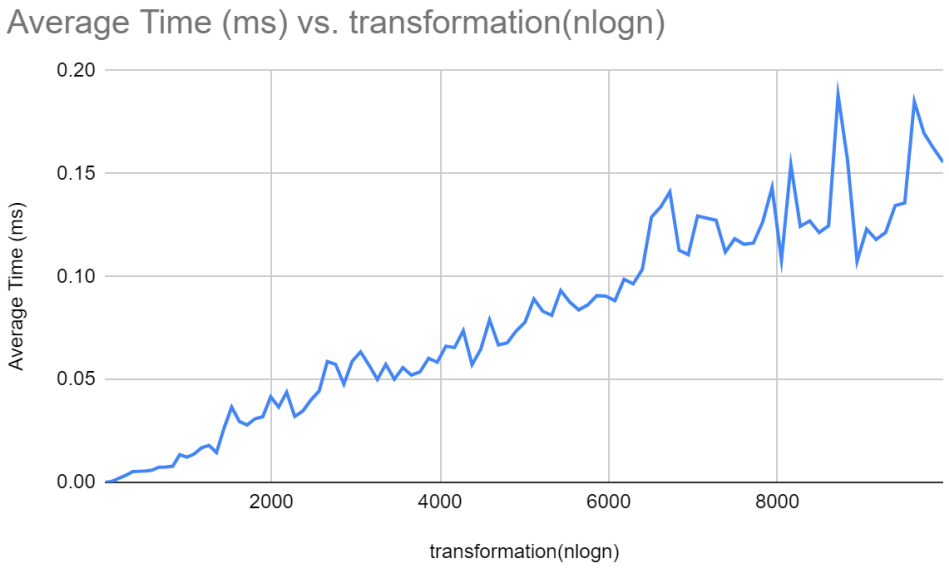
We can see that after the transformation the graph looks a little more linear (not counting the outliers).

Results for Quick Sort:

Time is in milliseconds.



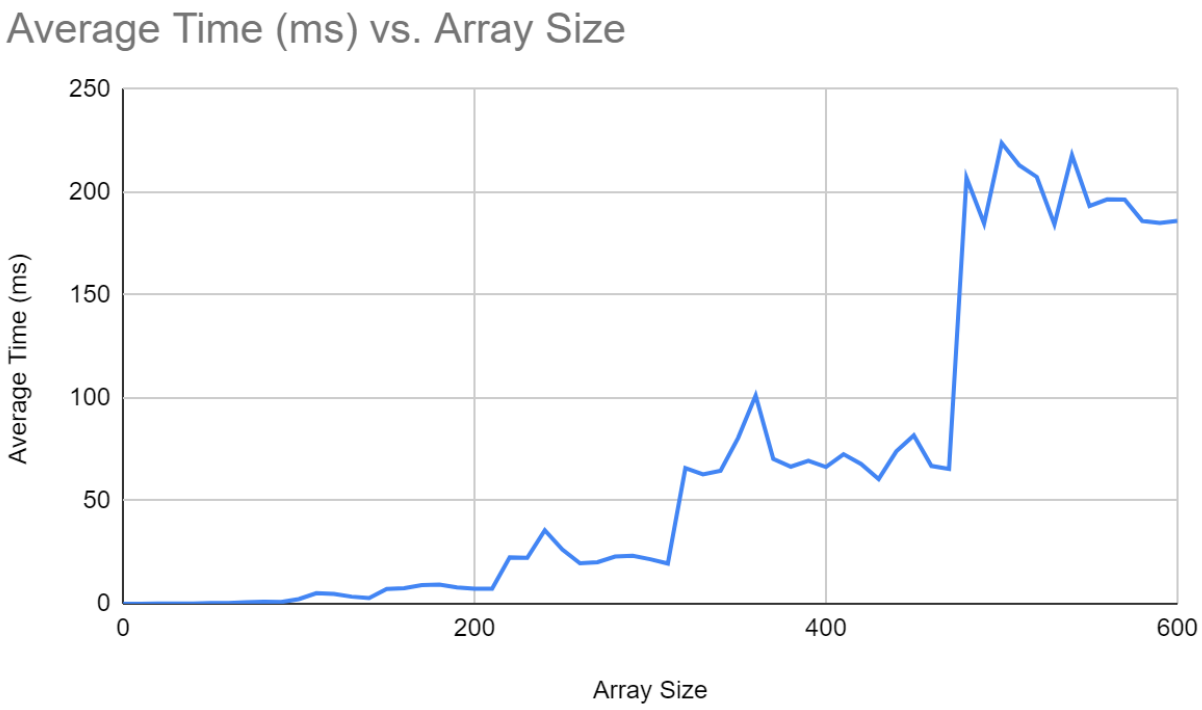
From this plot, we can kind of deduce that it is either linear or better time complexity. But since we know Quick sort has the worst case time complexity of nlog(n) we will do the transformation.



We can see that after the transformation the graph looks a little more linear (not counting the outliers).

Results for Stooge Sort:

Time is in milliseconds.



From this plot, we can come to the conclusion that it is worse than linear time complexity. It is interesting to see the jumps in the time at around 200, 300, 450. I am not really sure why these jumps are there. It could be because as the size gets really big the possibility of elements being somewhat reverse ordered gets higher.