1. Quick Sort versus Merge Sort

- Number of iterations per value of n: 500
- The unit of time is microseconds (μs)

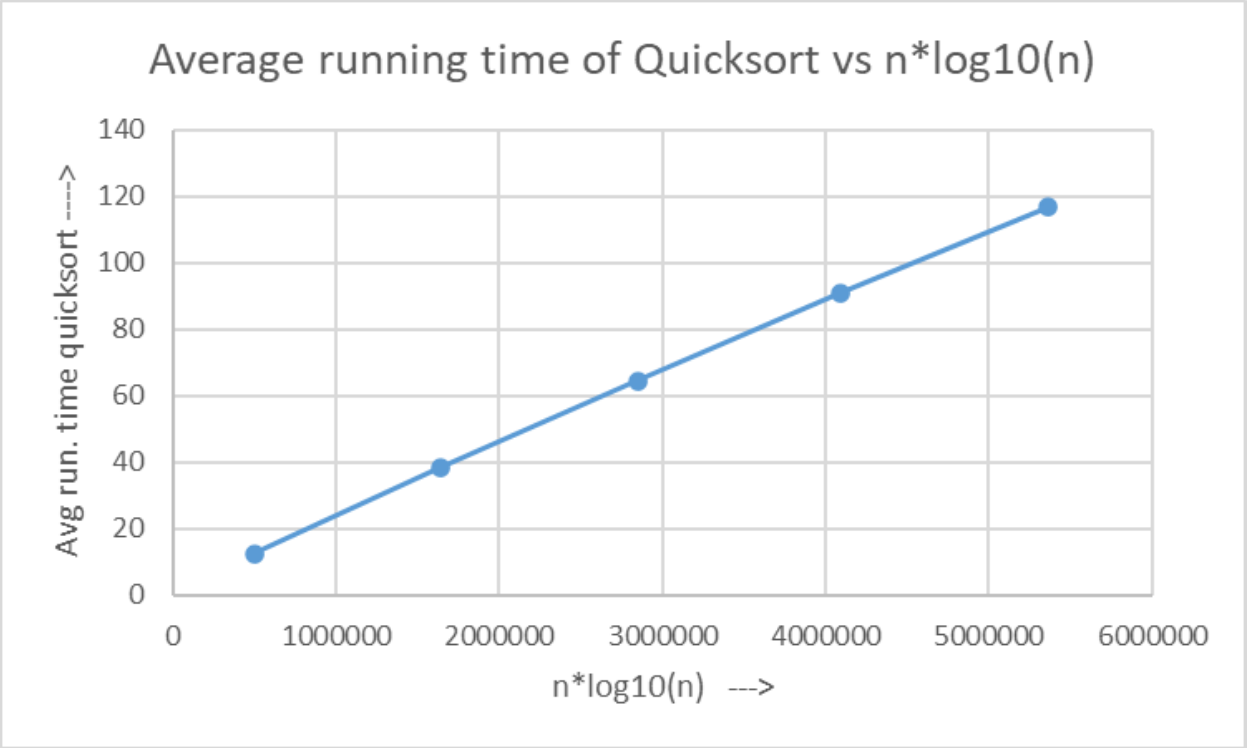
1.1 Comparisons

n →	10^2	10^3	10^4	10^5	10^6
Average number of comparisons during Quick Sort	948	14626	198778	2516357	27706761
$2n\log_e n$	921	13815	184206	2302585	27631021
Average number of comparisons during Merge Sort	542	8707	120450	1536391	18674192
$n\log_2 n$	664	9965	132877	1660964	19931568

From data, number of comparisons during quicksort grows somewhat linearly with $2*n*\log_e n$. The number of comparisons should be $O(n*\log_2 n)$. The time complexity of the merge sort is $O(n*\log_2 n)$, as the time taken by the merge sort is dependent on the number of comparisons.

1.2 Number of comparisons and time complexity of the quick sort

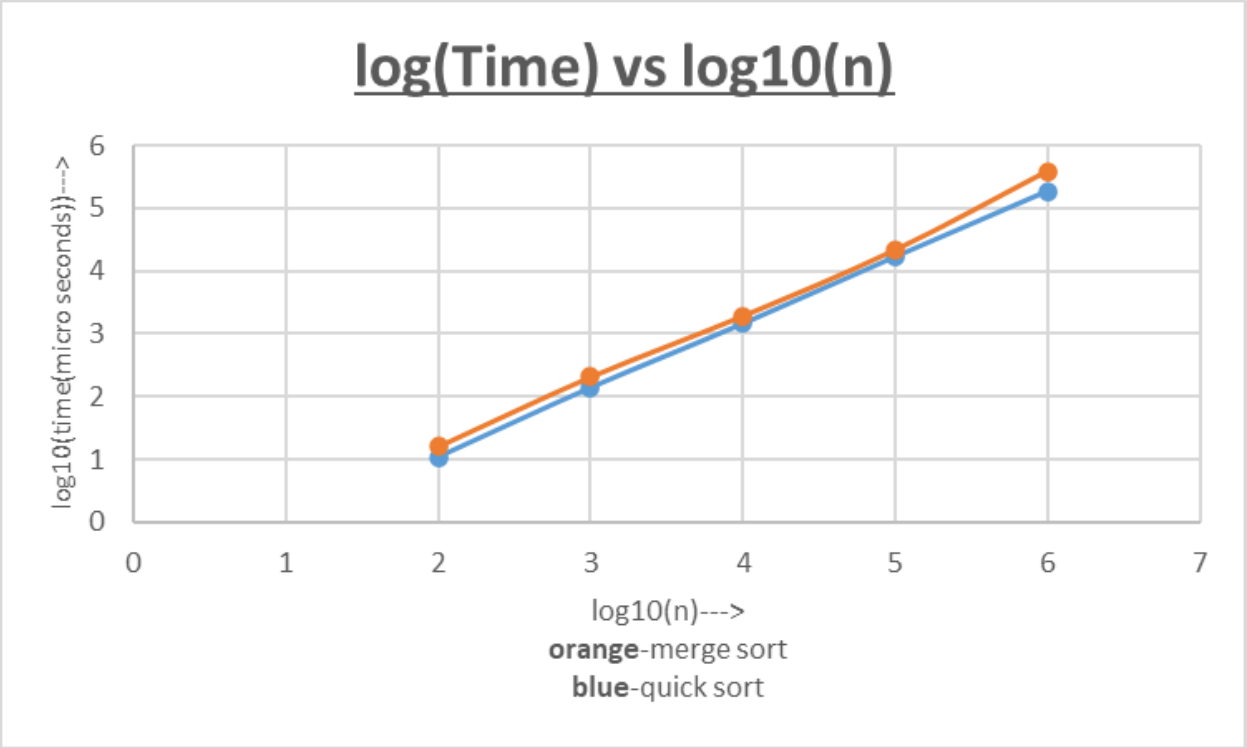
n→	10^5	$3 * 10^5$	$5 * 10^5$	$7 * 10^5$	$9 * 10^5$
Average running time of Quick Sort	16684.954000	53952.552000	94458.568000	131232.292000	132459.640000



As the graph is linear, so the average running time of quicksort will be $O(n*\log_{10}n)$. The graph matches the theoretical observations.

1.3 Time Complexity

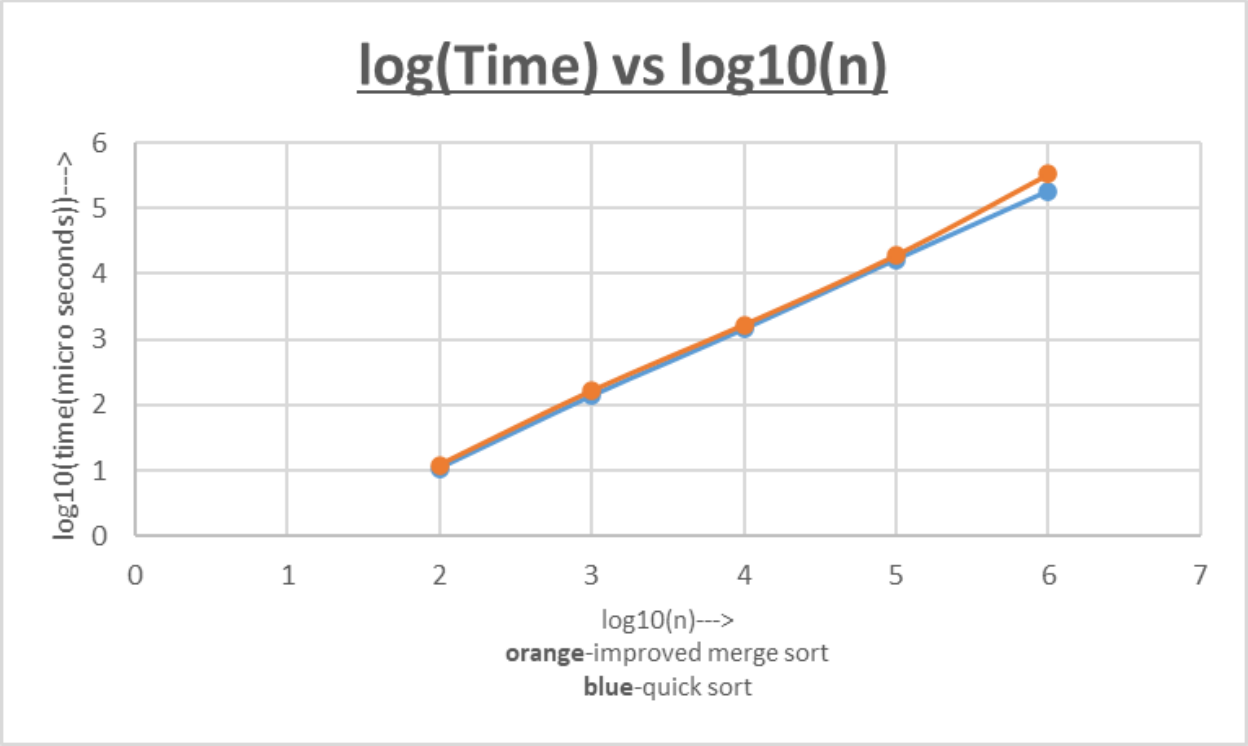
n→	10^2	10^3	10^4	10^5	10^6
Average running time of Quick Sort	10.884000	138.044000	1441.302000	16684.954000	185685.188000
Average running time of Merge Sort	16.190000	205.776000	1904.126000	21339.136000	379952.524000
Number of times Merge Sort outperformed Quick Sort	4	0	2	0	0



Graph shows that the time taken by quick sort is less than that of merge sort and also both the plots are linear which suggest that both merge sort and quick sort have $O(n*\log_{10}n)$ average time.

1.4 Can you improve merge-sort?

n→	10^2	10^3	10^4	10^5	10^6
Average running time of Quick Sort	10.884000	138.044000	1441.302000	16684.954000	185685.188000
Average running time of improved Merge Sort	11.918000	162.346000	1656.884000	19273.028000	334232.844000
Number of times Improved-Merge Sort outperformed Quick Sort	8	4	26	10	0



Quick-sort is still faster than Improved merge sort in terms of average time taken. This is more observable when the input size is large.

2. Reliability of Quick Sort

n→	10^2	10^3	10^4	10^5	10^6
Average running time of Quick Sort	10.884000	138.044000	1441.302000	16684.954000	185685.188000
No. of cases where run time exceeds the average by 5%	115	49	39	37	9
No. of cases where run time exceeds the average by 10%	102	49	32	14	2
No. of cases where run time exceeds the average by 20%	24	13	13	20	0
No. of cases where run time exceeds the average by 30%	12	6	0	1	0
No. of cases where run time exceeds the average by 50%	11	0	0	0	0

No. of cases where run time exceeds the average by 100%	4	0	0	0	0

Quick sort is more reliable when the input size is large. We can see from the above table that for large inputs the cases when running time is exceeding are very less.