Complexity Analysis of Sorting Algorithms

Student Name: Zannatul Naim
Id: 1805024
Course No: CSE 204

Complexity analysis:

From the collected data, we can say that the time complexity of merge sort is O(nlogn). It’s because in every step, the array is divided into two arrays and those arrays are sorted in linear time. In all order the time complexity remains the same. The space complexity of merge sort is O(n) because we are copying the full input array.

For quick sort, in the random case , we can say that the time complexity is O(nlogn) But for ascending or descending order of array , we can see that the time of sorting is higher. Almost O(n*n) . This is because we set the pivot as the first value of the array. As descending or ascending order , the array is already sorted, so the recursion goes for n steps, and for each step it took n times to sort. This is why the time complexity is O(n*n) .

We can say that we can use merge sort for any order array. But quick sort works fine for array containing random order of elements.

Machine Configuration:

Processor: Intel® Core™ i5-8250U CPU @ 1.60GHz × 8
Ram: 8gb
OS: Ubuntu 18.04.5 LTS

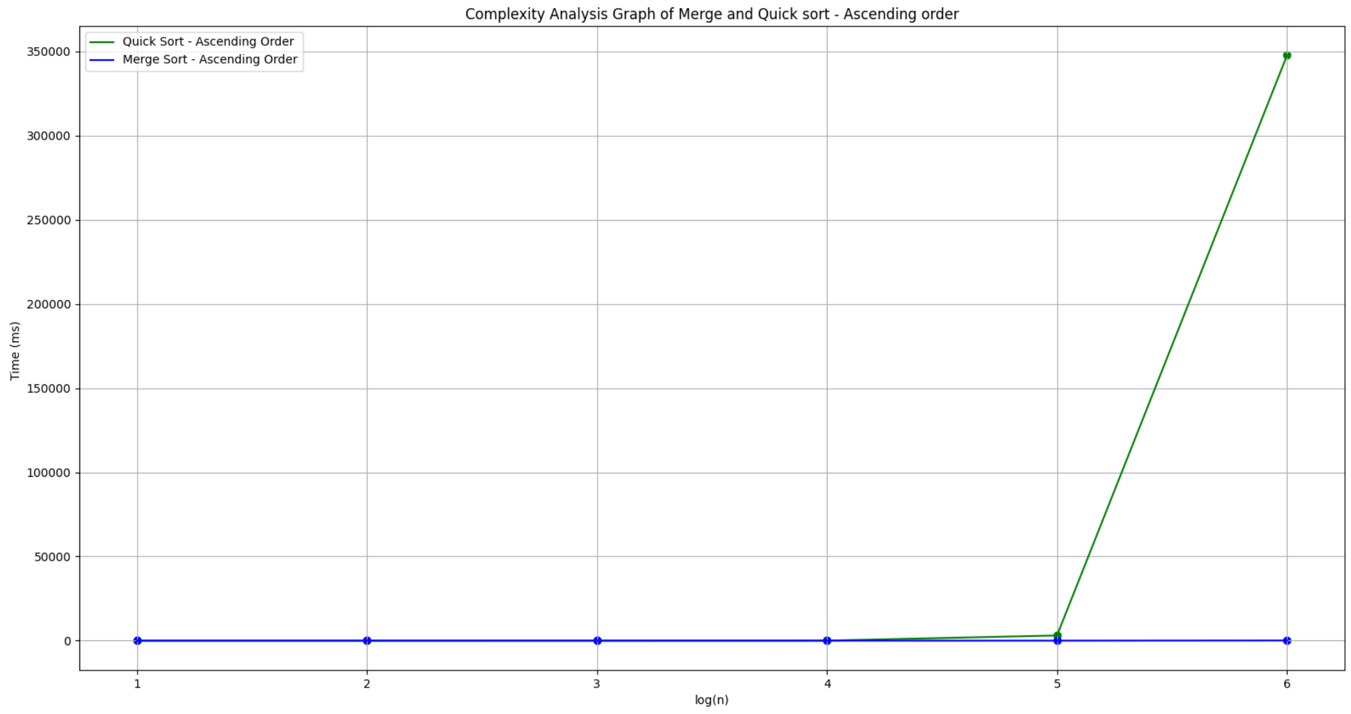
Table:

Input order	n=	10	100	1000	10000	100000	1000000
	Sorting Algorithm						
Ascending	merge	0.6	0.1	0.81	2.6	14.42	103.68
	quick	0.53	0.11	2.5	54.94	3089.5	347609.6
Descending	merge	0.008	0.08	0.29	1.15	20.15	105.5
	quick	0.003	0.16	2.7	37.69	2848.24	340344.5
Random	merge	0.01	0.1	0.45	1.9	22.7	195.23
	quick	0.005	0.04	0.26	1.06	15.86	158.59

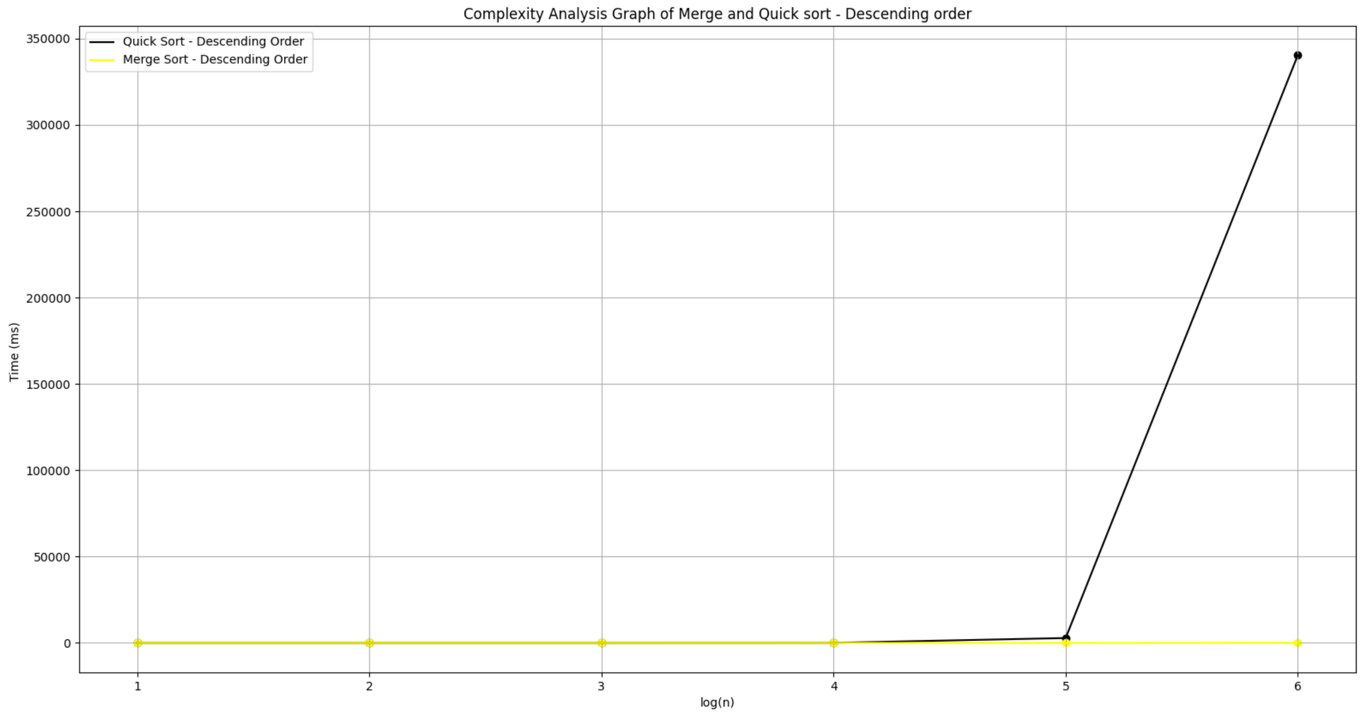
Table: Average time for sorting n integers in different input orders

Graph:

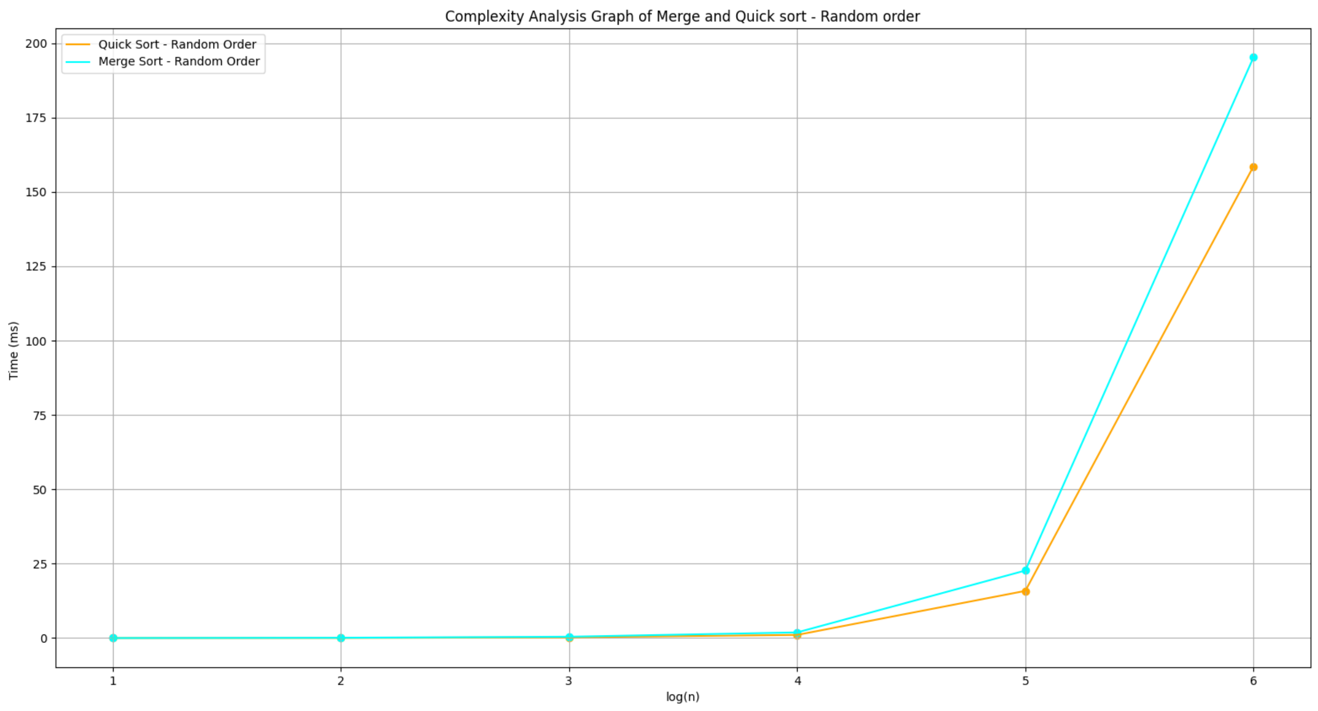
For ascending order:



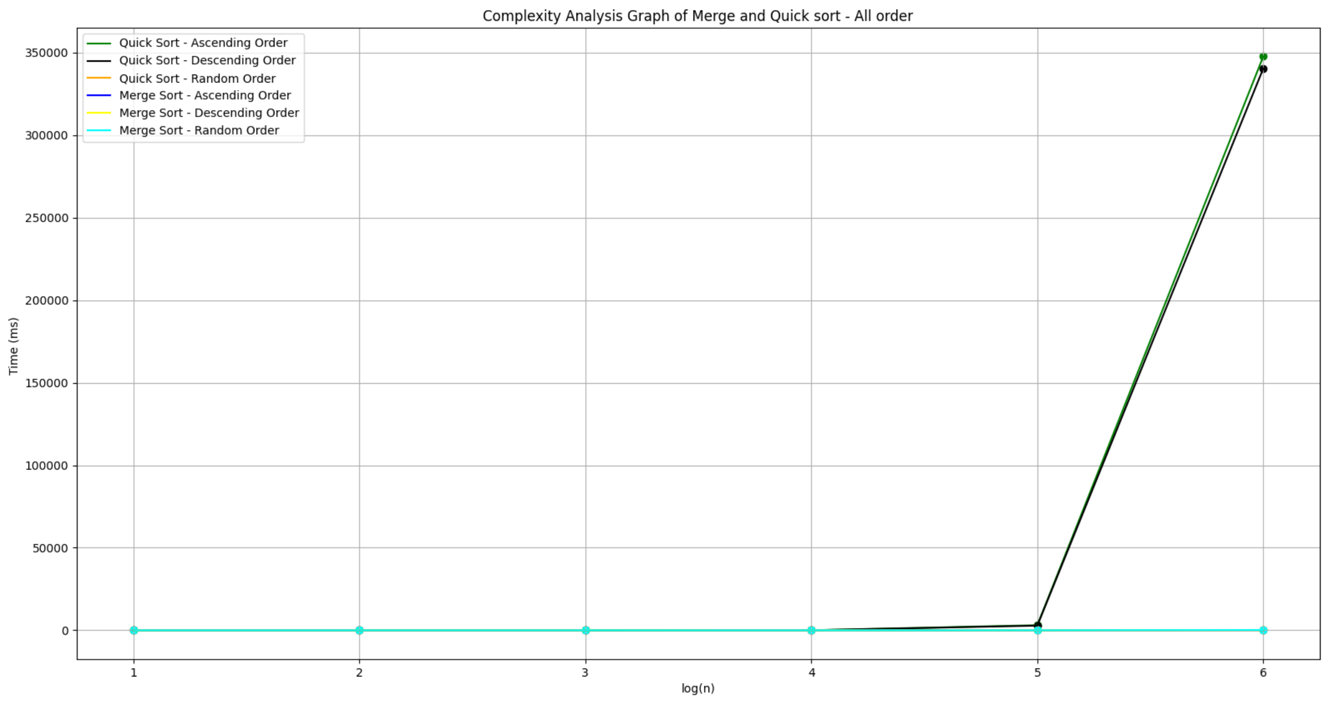
For descending order:



For random order:



For all order:


Complexity Analysis Graph of Merge and Quick sort - All order
Quick Sort - Ascending Order
Quick Sort - Descending Order
Quick Sort - Random Order
Merge Sort - Ascending Order
Merge Sort - Descending Order
Merge Sort - Random Order
Time (ms)
log(n)