1081 Algorithms
Programming Assignment #1: Sorting

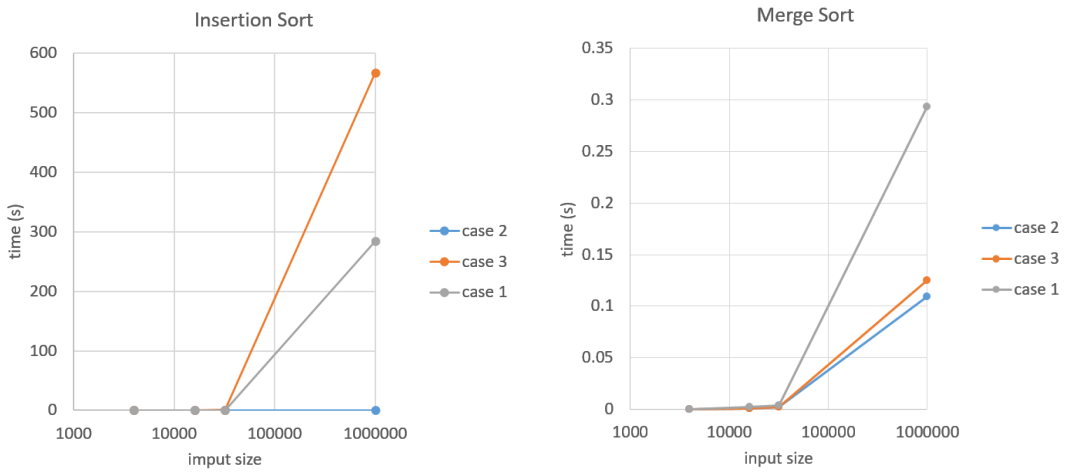
電機三 張恆瑞
B06901020

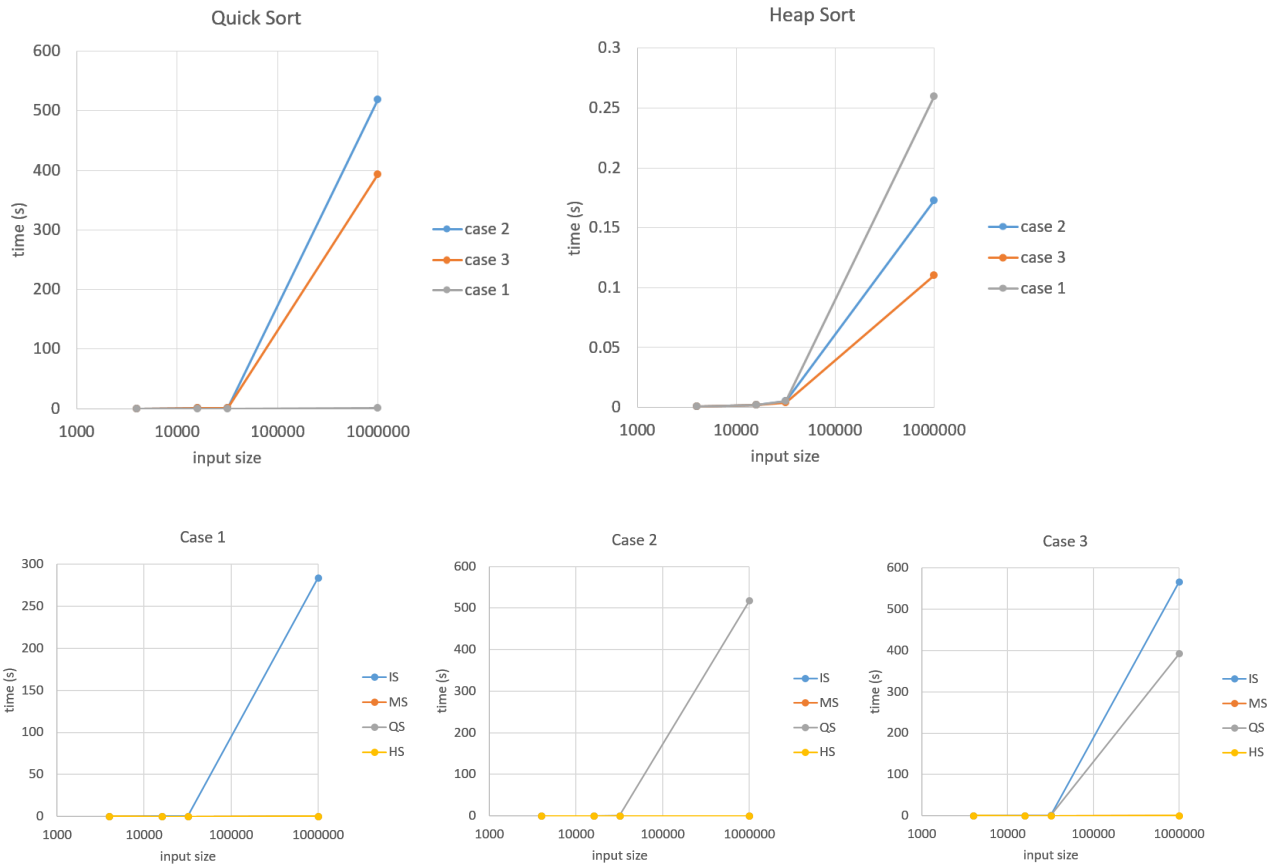
1 Running Time

All results are averaged by 5 repeated experiments conducted on Intel Xeon E5620@2.4GHz on EDA Union Workstation. And ”Time” below stands for ”CPU time”.

Input size	Insertion Sort		Merge Sort		Quick Sort		Heap Sort	
	Time (s)	Memory (KB)	Time (s)	Memory (KB)	Time (s)	Memory (KB)	Time (s)	Memory (KB)
4000.case2	0	12508	0	12508	0.016	12508	0.001	12508
4000.case3	0.016	12508	0	12508	0.014	12508	0.001	12508
4000.case1	0.008	12508	0	12508	0.001	12508	0.001	12508
16000.case2	0	12656	0.001	12656	0.242	12656	0.002	12656
16000.case3	0.241	12656	0.001	12656	0.220	12816	0.002	12656
16000.case1	0.120	12656	0.002	12656	0.003	12656	0.002	12656
32000.case2	0	12656	0.002	12708	0.829	12656	0.005	12656
32000.case3	0.911	12656	0.002	12708	0.715	13052	0.004	12656
32000.case1	0.289	12656	0.004	12708	0.003	12656	0.005	12656
1000000.case2	0.004	18676	0.109	20464	518	18676	0.173	18676
1000000.case3	567	18676	0.125	20464	393	25392	0.110	18676
1000000.case1	284	18676	0.293	20464	0.167	18676	0.260	18676

2 Rate of Growth





Note that, case 1: random ordered sequence, case 2: sorted sequence, case 3: reverse ordered sequence

3 Discussion

1. It takes **insertion sort** significantly more time on cases 1 and 3, which is the worst case of time complexity $O(n^2)$. However, it is efficient when the sequence is sorted beforehand, since in this case, this algorithm does not need to insert any element during the process, resulting in an $O(1)$ operation for each loop, i.e., an $O(n)$ in total.

2. Apparently, **merge sort** has similar results for all three cases, which are all much faster than the previous discussed insertion sort. This experiment verified the theoretical time complexity $\Theta(n \log n)$ of merge sort.

3. **Quick sort** has the shortest run time among the conducted four sorting algorithm experiments on the average case, which verified the $O(n \log n)$ complexity on average cases. Nevertheless, this does not apply to the other two cases. Quick sort has worst cases when the sequence is sorted or reversed as we apply the *Hoare partition*, which results in an $O(n^2)$ complexity.

4. **Heap sort** does also a great job on all cases, which is similar to merge sort, an $\Theta(n \log n)$ time complexity.

5. Overall, among the four sorting algorithms, we conclude that: (a) quick sort performs the best on average cases, (b) insertion sort performs the worst on average cases, (c) merge sort and heap sort are very stable in time complexity($\Theta(n \log n)$).