Computer & Systems Engineering Department

Data Structures and Algorithms

Implementing Binary Heap & Sorting Techniques

Contributors:

	Name	ID
1	Adel Mahmoud Mohamed Abdelrahman	20010769
2	Amr Ahmed Abdelazim	20011037
3	Marwan Essam Eldin	20011859
4	Mohamed Amr Abdelfattah	20011675
5	Mennat-Allah Mahmoud Saad	19016713

Time and Space Complexity of the Sorting algorithms:

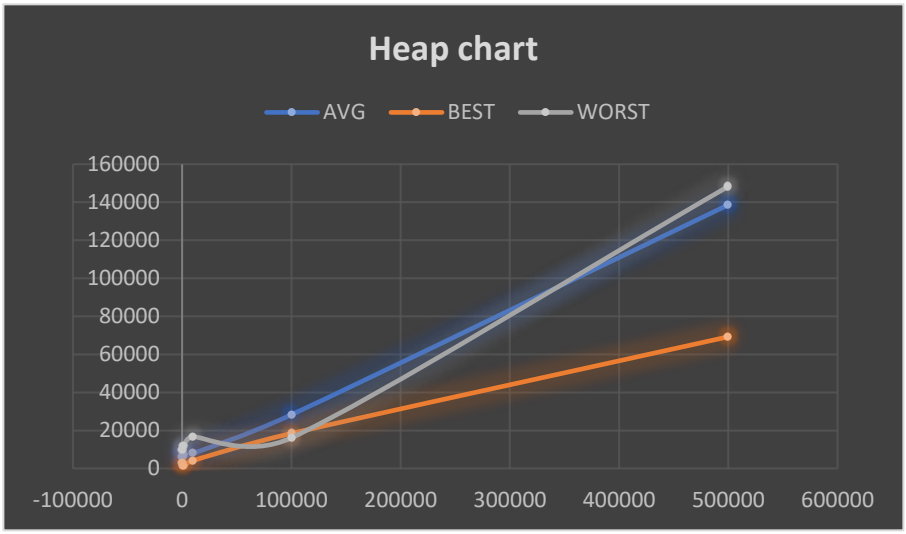
Algorithm	*Space complexity*	*Time Complexity*
Counting sort	***O (N + K)***	***O(N + K)***
Merge sort	***O(N)***	***O (N Lg(N))***
Heap sort	***O(N)***	***O (N Lg (N))***
Bubble sort	***O (1)***	***O (N^2)***

Comparison between them according to the mean time to get the array sorted:

Heap sort:

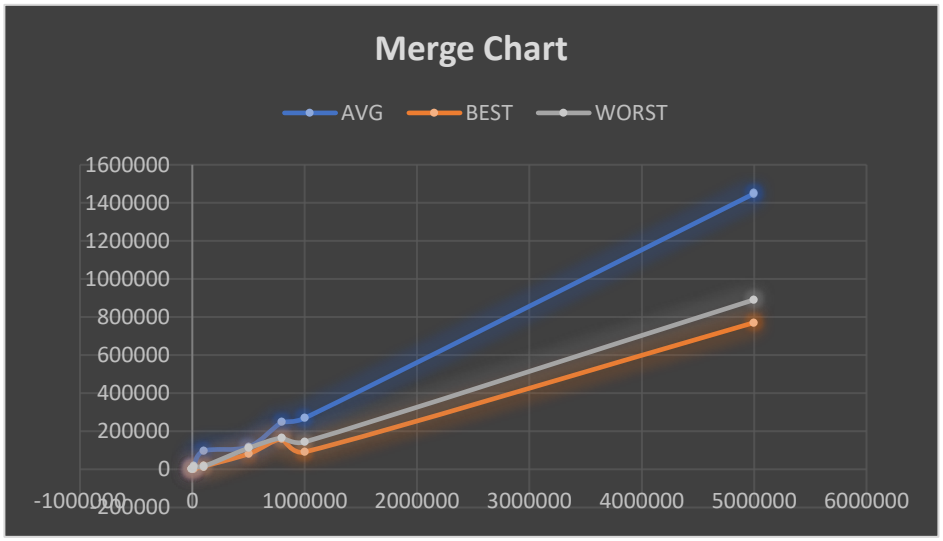
- **AVG: random elements.**
- **Best: Already sorted in a reverse order (which in our case (max heap) descending order).**
- **Worst: When it's sorted ascendingly.**

HEAP	AVG	BEST	WORST
100	6000.5	2731.7	9694.8
1000	6998.9	1465.9	11627.7
10000	7991.6	4143.6	16741
100000	28267.3	18434.8	16111
500000	138665.8	69260.7	148332



- **Merge sort:**
  - **AVG: random elements.**
  - **Best: Already sorted in Ascendingly.**
  - **Worst: When it's sorted discerningly.**

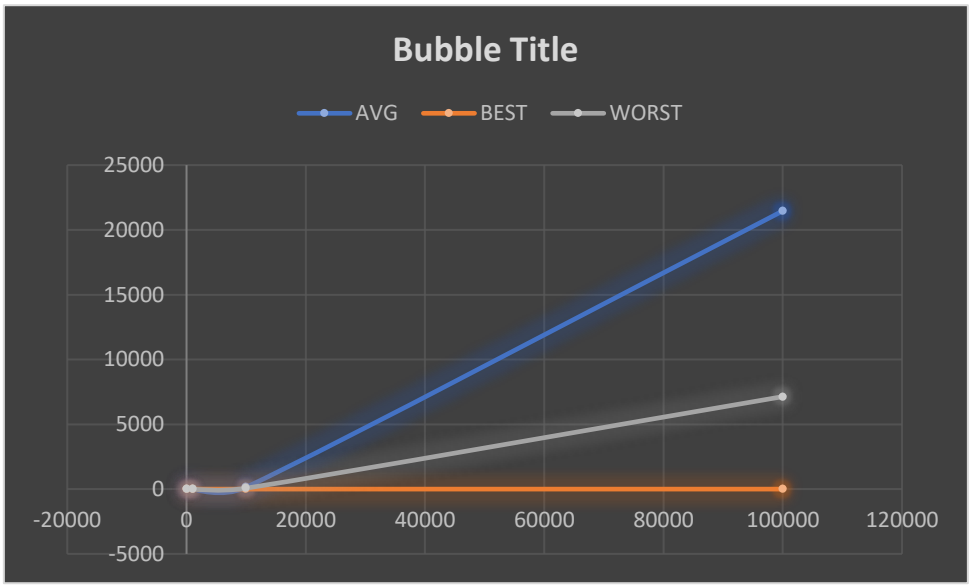
Merge	AVG	BEST	WORST
100	78.7	54.1	50.5
1000	350.7	298.8	691.2
10000	3748.2	1793.5	15452.2
100000	96379.2	16268	16356.9
500000	117842.4	81790.5	111814.9
800000	248391.2	160190.6	165734.7
1000000	269323.7	91707.1	144137.2
5000000	1448516	771022.4	891245.2
Mean	724297.5	385538.3	445647.9



- **Bubble sort:**

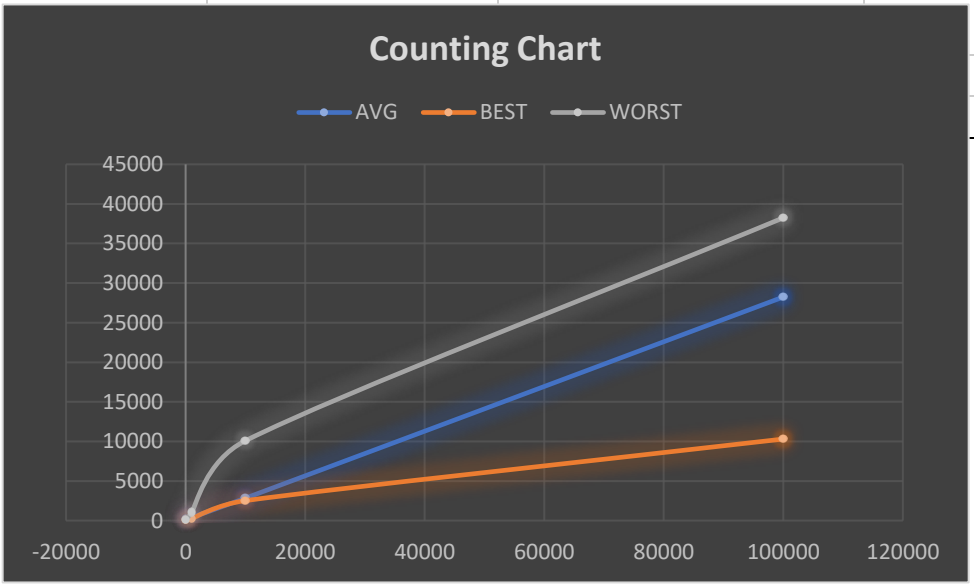
WITH Millis

Bubble	AVG	BEST	WORST
100	1	0	0
1000	5	0	1
10000	144	0	72
100000	21463.5	0	7136
Mean	10732.25	0	3568



- **Counting sort**
  - **AVG: random elements.**
  - **Best: when the range of the elements is small K is small, and the elements are repeated frequently like the case of having 100,000 elements from 0 to 9!**
  - **Worst: when the date is so scattered that it enlarges K so much.**

Counting	AVG	BEST	WORST
100	174.5	82.1	141.3
1000	405.1	243.3	1006.1
10000	2828.7	2520.9	10083.9
100000	28242.8	10299.8	38243.9
Mean	174.5	82.1	141.3



	Heap	Counting	Merge	Bubble
100	6000.5	174.5	78.7	1000
1000	6998.9	405.1	350.7	5000
10000	7991.6	2828.7	3748.2	144000
100000	28267.3	28242.8	96379.2	21463500
500000	138665.8		117842.4	
800000	224709.3		248391.2	
1000000	261773.5		269323.7	
5000000	18411110		1448516.3	
Mean	1680.689613	7.912775	273.0788	5403.375

Conclusion:

- The counting sort is the fastest algorithm, but it depends on the range of numbers.
- The second one is the merge sort algorithm and more stable than counting sort.
- after that, the heap sort comes after merge sort because Merge sort has a more predictable and consistent performance due to its divide-and-conquer approach, and have direct access to the arrays without performing the swapping operation, while heap sort involves more operations -such as swapping- in maintaining the heap property, and this introduce a constant factor to the complexity of the heap sort which is approximately 4.
- the slowest one is bubble sort as it's expected O(N2), it didn't even pass the test cases larger than 100,000.