One liners for Sorting Algorithms

1. Bubble Sort

- It works by repeatedly swapping the adjacent elements if they are in wrong order.
- O(n*n). Worst case occurs when array is reverse sorted.
- It can be optimized by stopping the algorithm if inner loop didn't cause any swap.
- Hence Best case improves to O(n) occurs when array is already sorted.

2. Selection Sort

- It sorts an array by repeatedly finding the minimum element (considering ascending order) from unsorted part and putting it at the beginning.
- O(n*n) worst case

3. Insertion Sort

- We maintain two subarrays in array first one is sorted and other unsorted
- Each time we pick element from unsorted part and find its correct position in sorted part.
- Hence each iteration the size of sorted part is increased by one and size of unsorted get decreased by one.
- Best Case – O(n) when array is sorted
- Worst Case – O(n*n) when array is sorted in descending order

4. Merge sort

- Merge sort is a divide-and-conquer algorithm.
- We recursively break our problem(array) into subparts until subpart become a single element.
- Then we merge subparts to make sorted array.
- O(n*n) For all cases
    i. Dividing takes logn time
    ii. Merging takes O(n) time

5. Quick Sort

- Quick sort is based on the divide-and-conquer approach
- We chose a pivot and find its right index using partition algorithm
- The partitioning is done such that Left side of pivot contains all the elements that are less than the pivot element Right side contains all elements greater than the pivot.
- Worst Case : O(n*n) when array is sorted in ascending or descending order
- Average case : O(n log n)

6. Heap Sort

- Heap sort uses heap data structure. We can either use MIN heap or MAX heap.

- Suppose if we use MAX heap. Then we first convert array into MAX heap using heapify.
- Then we pick root of heap (max element) and put it in end of array.
- Again we repeat this process until heap is empty and we pick max element and put in second last, then third last and so on
- O(nlongn) in all cases.

<u>Important Questions</u>

1. **Why Quick Sort is preferred over MergeSort for sorting Arrays?**
2. **Why MergeSort is preferred over QuickSort for Linked Lists?**
3. **Which is best sorting algorithm when array is almost sorted.**
4. **When does Quick sort worst case occurs?**
5. **What are inplace sorting algorithm with example?**