1. **What are the advantages and disadvantages of merge and quick sort?**

Merge Sort:

Advantages:

- It is used in sorting high memory data files and sorting huge amounts of files.
- It doesn't change the order of the previous data. It sorts the data in the right order and is stable.
- In merge sort, we can merge two linked lists by changing their nodes, addresses, and pointers without creating a third linked list .
- Merge sort executes direct merging of sorted subarrays and has a constant speed on any size of data.

Disadvantages:

- It takes greater time for shorter algorithms compared to the other sorting methods since it is suitable for larger algorithms.It takes smaller time for larger algorithms in contrast to others.
- Its running time is 0 and it takes additional memory space in sorting algorithms.
- It requires twice the memory of the heap sort due to the second array and performs the whole process despite the list is sorted

Quick Sort:

Advantages:

- In quick sorting, the algorithm does not require extra storage space. The sorting takes place in the given area and covers the exact storage to the unsorted list. Quick sort has the advantage over other algorithms in the same category. Moreover, the quick sort algorithm is quicker in an environment such as virtual memory since it demonstrates good cache locality.
- It enables users to obtain the same result faster than any other sorting algorithms. Using this algorithm, users could choose any value from the list, which will be imitated as a pivot.
- Time and space complexity. It has a space complexity of 0(log n) and has the best time complexity compared to other sorting algorithms.

Disadvantages:

- It has a worst-case time complexity. It typically happens when the element chosen as the pivot is the smallest or largest element and when all the elements are exactly the same. These worst cases affect the performance of the quick sort.
- It is unstable because it can't maintain the order of the elements while reordering.
- The implementation is complex if recursion is unavailable since it is recursive

2. How to compute the time of merge and quick sort.

In **Merge Sort**, it is fast and has a time complexity of `O(n*log n)`. Merge sort divides the array in two halves and takes linear time to combine two halves.Also, it is a stable sort. It implies that the equal elements are arranged in the same order in the sorted list.

- Best Case Time Complexity: **O(n*log n)**

- Worst Case Time Complexity: **O(n*log n)**

- Average Time Complexity: **O(n*log n)**

In **Quick Sort**, it follows a divide and conquer method. Recursion is typically used in the implementation of quick sort.

- Best Case Time Complexity : O(nlogn)

- Average Time Complexity : O(nlogn)

- Worst Case Time Complexity : O(n^2)

- Worst Case will happen when array is sorted

- Partition of elements take n time

- Problem is divide by the factor 2

References:
[1] **https://dev.to/ayabouchiha/merge-sort-algorithm-1l55**
https://www.techquintal.com/advantages-and-disadvantages-of-quick-sort/

[2] **https://www.geeksforgeeks.org/quick-sort-vs-merge-sort/**
https://www.mygreatlearning.com/blog/quick-sort-algorithm/