MUKIIZA ANGEL NINA TWINE S22B23/007

I) Terms and their relationship to searching and sorting

Linear Search:

Linear search is a simple searching algorithm that involves iterating through a list of elements one by one until a target element is found. It's often used to find a specific value in an unsorted or sorted list. The time complexity of a linear search is O(n) in the worst-case scenario, where 'n' is the number of elements in the list.

Binary Search:

Binary search is an efficient searching algorithm specifically designed for sorted lists. It works by repeatedly dividing the search interval in half and eliminating half of the remaining elements until the target element is found or the search interval becomes empty. Binary search has a time complexity of O(log n) in the worst-case scenario for sorted lists, where 'n' is the number of elements.

Merge Sort:

Merge sort is a divide-and-conquer sorting algorithm that divides a list into smaller sublists, sorts them, and then merges them back together. It has a consistent time complexity of O(n log n) for both the average and worst-case scenarios, making it an efficient sorting algorithm for larger datasets. Merge sort is a stable sort, which means it maintains the relative order of equal elements.

Quick Sort:

Quick sort is another divide-and-conquer sorting algorithm that works by selecting a "pivot" element and partitioning the list into two sublists, one with elements less than the pivot and one with elements greater than the pivot. These sublists are then recursively sorted. Quick sort has an average time complexity of O(n log n) and is often faster in practice than merge sort. However, its worst-case time complexity can be O(n^2), but this can be mitigated with proper pivot selection strategies.

II)Time Complexity Analysis:

- The time complexity of this function is O(n), where 'n' is the number of elements in the array. This is because it iterates over the entire array once, performing a constant amount of work for each element.

Space Complexity Analysis:The space complexity is O(1) because the function uses a constant amount of additional space, regardless of the size of the input array. It only stores a few variables, such as `max\_val`, which do not depend on the size of the array.