❖ Analysis of comparison based techniques.

1. Quick Sort:

- **In Quick sort comparison is done by pivot element, so quick sort performance is depend on which element you select as pivot element. Best case for quick sort is when most of the all the elements are sorted initially,for average case the sequence is randomly defined but it also has same time complexity as best case.In Worst case of the Quick sort is when the pivot element is placed in the table where in one side there is no element while in other there is actually n-1 elements. At that time the time taken by algorithm is increase which is order of n^2**

2. Heap Sort:

- **In Heap sort algorithm,first we have to create heapify so no matter in which order the element is come it must be inserted in heap,and then we have to repeatly takeout the root element from the heap.so the time complexity of heap sort in best,average and in worst case is same which is order of "nlogn".**

3. Merge Sort:

- **In merge sort,first we have to partition the table into two or more pieces the after sorting both adjacent table, at last we get completely sorted table, here the elements doesn't effect very much so the time complexity is same as Heap Sort.**

4. Bubble Sort:

- **In bubble sort algorithm we have to for loop or we can say that one loop is inner loop of other, so the as usual the time complexity is order of "n^2", But when the list is already sorted than this complexity is reduced to order on "n" and this is the best case of bubble sort.**

5. **Insertion Sort:**

- **This is the simple sorting algorithm which we used in our real life,that is compare the element with its previous element if the previous element is larger than this than move it towards next,here in this the worst case and average case has same complexity which order of "n^2" but when the element is initially sorted than this is same as linear time complexity of the algorithm.**

6. **Selection Sort:**

- **In the selection sort the approach is select the minimum key from the unsorted elements and put that key to the starting position of unsorted array,here no matter in which sequence the elements are come we have to find the smallest element from the unsorted array so time complexity of the selection sort is same in best,average and worst case which order of "n^2".**

❖ **Table Of Time Complexity**

Algorithm	Worst Case	Average Case	Best Case
Quick Sort	**$O(n^2)$**	**O(nlogn)**	**O(nlogn)**
Heap Sort	**O(nlogn)**	**O(nlogn)**	**O(nlogn)**
Merge Sort	**O(nlogn)**	**O(nlogn)**	**O(nlogn)**
Bubble Sort	**O(n)**	**$O(n^2)$**	**$O(n^2)$**
Insertion Sort	**O(n)**	**$O(n^2)$**	**$O(n^2)$**
Selection Sort	**$O(n^2)$**	**$O(n^2)$**	**$O(n^2)$**