LAB SESSION 11: SEARCHING AND SORTING

AIM: To implement the various searching and sorting algorithms.

PROBLEM DEFINITION:
Develop a C program to implement the following searching and sorting algorithms on arrays.

1. Linear Search
2. Binary Search
3. Bubble sort.
4. Selection sort.
5. Insertion sort.
6. Merge sort.

THEORY: Sequential Search is performed in a linear way i.e. it starts from the beginning of the list and continues till we find the item or reach the end of the list. The item to be searched is compared with each element of the list one by one, starting from the first element. The number of comparisons required to search for an item depends on the position of the element inside the array. The best case is when the item is present at the first position and in this case only one comparison is done. The worst case occurs when the item is not present in the array and in this case n comparisons are required where n is the total number of elements. Searching for an item that is present at the ith position requires i comparisons..

BINARY SEARCH:
The prerequisite for binary search is that the array should be sorted. Firstly we compare the item to be searched with the middle element of the array. If the item is found there, our search is finished successfully. Otherwise the array is divided into two halves, the first half contains all elements to the left of the middle element and the other one consists of all the elements to the right side of the middle element. Since the array is sorted, all the elements in the left half will be smaller than the middle element and the elements in the right half will be greater than the middle element. If the item to be searched is less than the middle

element, it is searched in the left half otherwise it is searched in the right half. Now search proceeds in the smaller portion of the array and the item is compared with its middle element.If the item is same as the middle element, search is finished otherwise again the subarray is divided into two halves and the search is performed in one of these halves. This process of comparing the item with the middle element and dividing the array continues till we find the required item or get a portion which does not have any element.

Bubble sort Algorithm

Bubble sort works on the repeatedly swapping of adjacent elements until they are not in the intended order. It is called bubble sort because the movement of array elements is just like the movement of air bubbles in the water. Bubbles in water rise up to the surface; similarly, the array elements in bubble sort move to the end in each iteration.

Although it is simple to use, in bubble sort there may be many swaps in a single pass. It is not suitable for large data sets. The average and worst-case complexity of Bubble sort is O(n2), where n is a number of items.

Selection Sort Algorithm

In selection sort, the smallest value among the unsorted elements of the array is selected in every pass and inserted to its appropriate position into the array. It is also the simplest algorithm. In selection sort, the first smallest element is selected from the unsorted array and placed at the first position. After that second smallest element is selected and placed in the second position. The process continues until the array is entirely sorted. It is not suitable for large data sets.

Insertion Sort Algorithm

Insertion sort works similar to the sorting of playing cards in hands. It is assumed that the first card is already sorted in the card game, and then we select an unsorted card. If the selected unsorted card is greater than the first card, it will be placed at the right side; otherwise, it will be placed at the left side. Similarly, all unsorted cards are taken and put in their exact place.

The same approach is applied in insertion sort. The idea behind the insertion sort is that first take one element, iterate it through the sorted array. Although it is simple to use, it is not appropriate for large data sets as the time complexity of insertion sort in the average case and worst case is O(n2), where n is the number of items. Insertion sort is less efficient than the other sorting algorithms like heap sort, quick sort, merge sort, etc.

Merge Sort Algorithm

Merge sort is the sorting technique that follows the divide and conquer approach. It is one of the most popular and efficient sorting algorithm. It divides the given list into two equal halves, calls itself for the two halves and then merges the two sorted halves. We have to define the merge() function to perform the merging.

The sub-lists are divided again and again into halves until the list cannot be divided further. Then we combine the pair of one element lists into two-element lists, sorting them in the process. The sorted two-element pairs is merged into the four-element lists, and so on until we get the sorted list.

ALGORITHM WITH ILLUSTRATION (STEP BY STEP SOLVED EXAMPLE):

1. **Linear Search**
2. **Binary Search**
3. **Bubble sort.**
4. **Selection sort.**
5. **Insertion sort.**
6. **Merge sort.**