SE(ECE)	**Data Structures and Algorithms**	
Experiment No: 2	**Implement a) Linear Search and b) Binary Search algorithms using C++**	**Page: 1/3**

Aim of Experiment

To implement and compare Linear Search and Binary Search algorithms in C++ for searching an element from a list of data items.

Objectives

- To understand the concept of searching techniques.
- To implement Linear Search algorithm using C++.
- To implement Binary Search algorithm using C++.
- To analyze the working principle and efficiency of both algorithms.
- To compare the time complexity of Linear Search and Binary Search.

Theory

Searching

Searching is a process of finding whether a particular element is present in a given collection of data. Efficient searching algorithms are important for faster data retrieval in large datasets.

Linear Search

Linear Search (or Sequential Search) is the simplest searching technique where each element in the list is checked sequentially until the required element is found or the list ends.

Algorithm (Linear Search):

```
Step 1: Start
Step 2: Input array and element to be searched (key).
Step 3: Repeat for each element of array:
        If current element == key
            Return index (position found)
Step 4: If key not found, return -1.
Step 5: Stop
```

Time Complexity: O(n)
Best Case: O(1) (if element is at first position)
Worst Case: O(n) (if element is at last position or not present)

Binary Search

Binary Search is a searching technique that works on a sorted array by repeatedly dividing the search interval into half.

Algorithm (Binary Search):

```
Step 1: Start
Step 2: Input sorted array and key.
Step 3: Initialize low = 0, high = n-1
Step 4: Repeat until low <= high
        mid = (low + high) / 2
        If array[mid] == key → Return mid (position found)
        Else if key < array[mid] → high = mid - 1
        Else → low = mid + 1
Step 5: If not found, return -1.
Step 6: Stop
```

Time Complexity: O(log n)
Best Case: O(1) (if middle element is the key)
Worst Case: O(log n)

Applications

Linear Search Applications:

- Suitable for small datasets or unsorted data.
- Used when data is rarely searched or unsorted (e.g., attendance list scanning).
- Useful in real-time systems where simplicity is preferred over efficiency.

Binary Search Applications

- Searching in large sorted datasets (e.g., dictionary lookup, phone directory).
- Widely used in databases and search engines.
- Applied in problems like finding square roots, decision-making algorithms, and optimization problems.

Questions

Task-1
Manually trace the Binary Search algorithm on a small, sorted list.

i. Prepare your data
   Create a sorted (ascending) list of **8 distinct integers**. (Make your own numbers)
   Use 0-based indexing: indices go from 0 to 7.
   Choose one target in **left half** portion of the list (a value that exists in your list.)
ii. Trace Binary Search- Apply binary search algorithm to search target of your choice. Show all steps.

SE(ECE)	Data Structures and Algorithms	
Experiment No: 2	Implement a) Linear Search and b) Binary Search algorithms using C++	Page: 3/3

Task-2
Manually trace the Binary Search algorithm on a small, sorted list.

i. Prepare your data
   Create a sorted (ascending) list of **9 distinct integers**. (Make your own numbers)
   Use 0-based indexing: indices go from 0 to 7.
   Choose one target in **right half** portion of the list (a value that exists in your list.)
ii. Trace Binary Search- Apply binary search algorithm to search target of your choice. Show all steps.

References

1. Yedidyah Langsam, Moshe J Augenstein, Aaron M Tenenbaum – Data structures using C and C++ - PHI Publications (2nd Edition).
2. Ellis Horowitz, Sataraj Sahni- Fundamentals of Data Structures – Galgotia Books source.

Conclusion

A	C	O	T	Sign.