Lab Report on Binary Search

1. Report No:

01

2. Report Name:

Binary Search Algorithm in C++

3. Objective:

To implement the Binary Search algorithm in C++ and understand its working by analyzing how it efficiently searches an element in a sorted array using divide and conquer technique.

4. Algorithm:

Binary Search (Iterative Method) Algorithm Steps:

1. Start with two pointers: low at the beginning and high at the end of the array.
2. Calculate the middle index mid = (low + high) / 2.
3. Compare the middle element with the target:
   - If equal, return the index (element found).
   - If the target is greater than the middle element, search the right half (low = mid + 1).
   - If the target is smaller, search the left half (high = mid - 1).
4. Repeat steps 2-3 until low > high.
5. If not found, return -1.

C++ Program:

```
#include <iostream>
using namespace std;

// Function to perform Binary Search
int binarySearch(int arr[], int size, int target) {
    int low = 0;
    int high = size - 1;
    while (low <= high) {
        int mid = (low + high) / 2;

        // Displaying current search bounds
        cout << "Searching in range index " << low << " to " << high << endl;

        if (arr[mid] == target) {
            return mid; // Element found
        } else if (arr[mid] < target) {
            low = mid + 1; // Move to right half
        } else {
            high = mid - 1; // Move to left half
        }
    }

    return -1; // Element not found
}

int main() {
    int arr[] = {2, 4, 7, 10, 13, 18, 21, 26, 30};
    int size = sizeof(arr) / sizeof(arr[0]);
    int target;

    cout << "Enter the number to search: ";
    cin >> target;

    int result = binarySearch(arr, size, target);

    if (result != -1) {
        cout << "Element found at index: " << result << endl;
    } else {
        cout << "Element not found in the array." << endl;
    }

    return 0;
}
```

5. Sample Input and Output:

Sample Run 1:

Input:

Lab Report on Binary Search

```
Enter the number to search: 13
```

Output:

```
Searching in range index 0 to 8
Searching in range index 0 to 3
Searching in range index 3 to 3
Element found at index: 4
```

Sample Run 2:

Input:

```
Enter the number to search: 9
```

Output:

```
Searching in range index 0 to 8
Searching in range index 0 to 3
Searching in range index 0 to 1
Element not found in the array.
```

6. Conclusion:

Binary Search is an efficient algorithm to search for an element in a sorted array.

Unlike linear search which takes O(n) time, Binary Search works in O(log n) time, making it suitable for large datasets.

The key requirement is that the array must be sorted. This algorithm demonstrates the power of the divide and conquer strategy

and forms the basis for many advanced algorithms in computer science.