Ahsania Mission University of Science & Technology

Lab Report

Course Code: CSE 2202

Course Title: Computer Algorithm Sessional

Experiment No: 05

Experiment Date: 12.03.25

Submitted By:

Mehrin Nusrat Chowdhury

Roll: 1012320005101026

1st Batch, 2nd Year, 1st Semester

Department of Computer Science and Engineering

Ahsania Mission University of Science & Technology

Submitted To:

Md. Fahim Faisal

Lecturer

Department of Computer Science and Engineering

Faculty of Engineering, Ahsania Mission University of Science & Technology

Submission Date: 19.03.25

Task: A C++ program that will implement the Binary Search algorithm.

Theory:

Binary search is a widely used algorithm for searching sorted arrays efficiently. It operates by repeatedly dividing the search interval in half, narrowing down the location of the target item. The process begins by comparing the middle element of the array to the search item. Based on the comparison, either the lower or upper half of the array is considered for the next iteration. This eliminates half of the potential search area in every step, making binary search significantly faster than linear search in terms of computational complexity. The method guarantees optimal performance for sorted datasets and is commonly employed in various applications like databases, dictionaries, and search engines. Its efficiency is marked by a time complexity of ($O(log\ n)$).

Code:

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

int main()
{
   int n;
   cout << "Enter the number of the elements: ";
   cin >> n;

   int a[n];
   cout << "Enter the elements: ";
   for(int i = 0 ; i < n ; i++)
   {
      cin >> a[i];
   }

   int item;
   cout << "Enter the Item: ";
   cin >> item;

   int b = 0;
   int e = n - 1;
   int mid = (b + e) / 2;
   int loc;

   while(b <= e && a[mid] != item)
   {
```

```
        if(item < a[mid])
        {
            e = mid - 1;
        }
        else
        {
            b = mid + 1;
        }

        mid = (b + e) / 2;
    }

    if(a[mid] == item)
    {
        loc = mid;
        cout << "Item found in location " << loc;
    }
    else
    {
        loc = NULL;
        cout << "Item not found in any location.";
    }

    return 0;
}
```

Output(s):

```
Enter the number of the elements: 10
Enter the elements: 12 34 45 66 67 78 79 81 89 95
Enter the Item: 89
Item found in location 8
Process returned 0 (0x0)   execution time : 53.430 s
Press any key to continue.
```

```
Enter the number of the elements: 10
Enter the elements: 12 34 45 66 67 78 79 81 89 95
Enter the Item: 52
Item not found in any location.
Process returned 0 (0x0)   execution time : 17.227 s
Press any key to continue.
```

Conclusion:

Through the implementation of binary search, we observed its systematic approach to locating elements in a sorted array. The program accurately demonstrated how the algorithm partitions the array into smaller intervals and checks the middle element iteratively. This study reinforced the understanding of binary search's effectiveness in optimizing search operations while reducing computational overhead. The results validated the algorithm's utility in situations requiring quick and efficient search techniques.