Name: Vaishnavi V. Poti

Batch: S21

Roll number: 2201078

**EX. 3:- IMPLEMENTATION OF BINARY SEARCH ALGORITHM**

LO Mapped:- LO1, LO2

Binary Search is defined as a searching algorithm used in a sorted array

by repeatedly dividing the search interval in half. The idea of binary

search is to use the information that the array is sorted and reduce the

time complexity to O(log N).

Conditions for when to apply Binary Search in a Data

Structure:

To apply Binary Search algorithm:

● The data structure must be sorted.

● Access to any element of the data structure takes constant time.

Binary Search Algorithm:

In this algorithm,

● Divide the search space into two halves by finding the middle index

“mid”.

●Compare the middle element of the search space with the key.

● If the key is found at middle element, the process is terminated.

● If the key is not found at middle element, choose which half will be

used as the next search space.

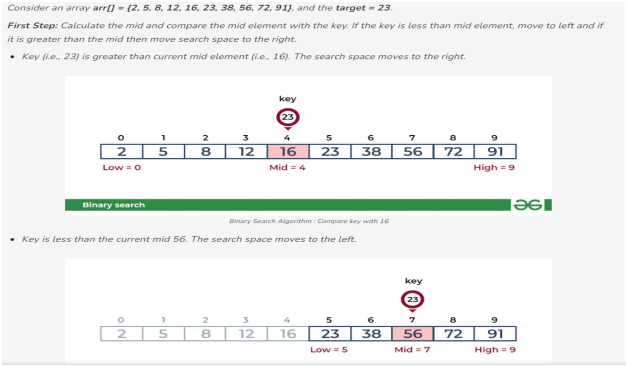
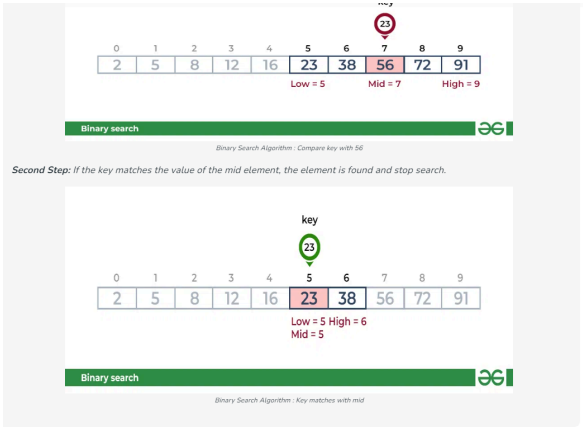
● If the key is smaller than the middle element, then the left

side is used for next search.

● If the key is larger than the middle element, then the right

side is used for next search.

● This process is continued until the key is found or the total search space is exhausted.



**CODE:**

//Binary Search

#include<stdio.h>

int binarySearch(int arr[], int l, int r, int x)

{

if(r >= l)

{

int mid = l + (r - l) / 2;

if (arr[mid] == x)

return mid;

if(arr[mid] > x)

return binarySearch(arr, l, mid - 1, x);

return binarySearch(arr, mid + 1, r, x);

}

return -1;

}

int main()

{

int x;

int arr[] = {2, 3, 17, 21, 42, 69, 71, 90, 97};

int n = sizeof(arr) / sizeof(arr[0]);

printf("Enter the element you want to find: ");

scanf("%d",&x);

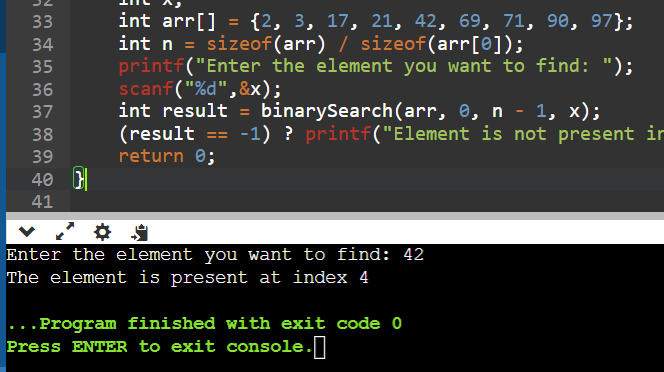
int result = binarySearch(arr, 0, n - 1, x);

(result == -1) ? printf("Element is not present in the array") : printf("The element is present at index %d", result);

return 0;

}

**OUTPUT:**



**CONCLUSION:**

We can conclude that we have successfully implemented Binary Search algorithm.