Name: Ashutosh Kumar

Roll Num: 231070006

COMPUTER ENGINEERING

DAA LAB ASSIGNMENT 2

PROGRAM:

LINEAR SEARCH :

```
#include <iostream>
#include <vector>
using namespace std;
int LinearSearch(vector<int> &numbers, int key)
{
    for (int i = 0; i < numbers.size(); ++i)
    {
        if (numbers[i] == key)
        {
            return i;  // key found at index i.
        }
    }
    return -1; // key not found.
}

int main()
{
    int n, key;
    // Get the size of the array
    cout << "Enter the number of elements in the array: ";
    cin >> n;
    // Create a vector and get the elements from the user
    vector<int> numbers(n);
    cout << "Enter " << n << " elements: ";
```

```
    for (int i = 0; i < n; ++i)
    {
        cin >> numbers[i];
    }
    // Get the key to search for
    cout << "Enter the number to search: ";
    cin >> key;
    // Perform Linear Search and output the result
    int result = LinearSearch(numbers, key);
    cout << (result != -1 ? "Number found at index: " + to_string(result) : "Number not found") << endl;
    return 0;
}
```

TEST CASE :

```
Enter the number of elements in the array: 5
Enter the elements of the array: 1 2 3 4 5
Enter the element to search for: 4
Element is present at index 3
```

```
Enter the number of elements in the array: 5
Enter the elements of the array: 1 7 9 16 12
Enter the element to search for: 9
Element is present at index 2
```

```
Enter the number of elements in the array: 5
Enter the elements of the array: 1 3 15 16 80
Enter the element to search for: 16
Element is present at index 3
```

```
Enter the number of elements in the array: 5
Enter the elements of the array: 1 2 5 7 9
Enter the element to search for: 6
Element is not present in array
```

```
Enter the number of elements in the array: 4
Enter the elements of the array: 3 5 7 11
Enter the element to search for: 15
Element is not present in array
```

PROGRAM:

BINARY SEARCH :

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

int binarySearch(int arr[], int left, int right, int x) {
    while (left <= right) {
        int mid = left + (right - left) / 2;
        // Check if x is present at mid
        if (arr[mid] == x)
            return mid;
        // If x is greater, ignore the left half
        if (arr[mid] < x)
            left = mid + 1;
        // If x is smaller, ignore the right half
        else
            right = mid - 1;
    }
    // If the element is not present, return -1
    return -1;
}
int main() {
    int n, x;
    // Get the size of the array from the user
    cout << "Enter the number of elements in the array: ";
    cin >> n;
    int arr[n]; // Declare the array with the user-defined size
    // Get the elements of the array from the user
    cout << "Enter the elements of the array in ascending order: ";
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }
```

```
    // Get the key to search for
    cout << "Enter the element to search for: ";
    cin >> x;
    // Perform the binary search
    int result = binarySearch(arr, 0, n - 1, x);
    if (result == -1)
        cout << "Element is not present in array" << endl;
    else
        cout << "Element is present at index " << result << endl;
    return 0;
}
```

TEST CASE :

```
Enter the number of elements in the array: 5
Enter the elements of the array in ascending order: 7 9 11 15 20
Enter the element to search for: 11
Element is present at index 2
```

```
Enter the number of elements in the array: 7
Enter the elements of the array in ascending order: 2 4 6 8 10 12 14
Enter the element to search for: 14
Element is present at index 6
```

```
Enter the number of elements in the array: 7
Enter the elements of the array in ascending order: 1 3 5 7 9 11 13
Enter the element to search for: 11
Element is present at index 5
```

```
Enter the number of elements in the array: 5
Enter the elements of the array in ascending order: 20 40 60 80 100
Enter the element to search for: 65
Element is not present in array
```

```
Enter the number of elements in the array: 1
Enter the elements of the array in ascending order: 1
Enter the element to search for: 6
Element is not present in array
```

Conclusion :

In this assignment, we focused on implementing Linear Search and Binary Search algorithms while adhering to best coding practices. We meticulously outlined the steps for each algorithm and developed test cases that covered both successful and unsuccessful search scenarios. Through analysis, we confirmed that Linear Search operates with a time complexity of O(n)O(n)O(n), whereas Binary Search operates with a time complexity of O(log⁡n)O(\log n)O(logn).

The assignment highlighted the efficiency of Binary Search when dealing with sorted data, emphasizing its advantages over Linear Search in such contexts. This exercise not only reinforced our understanding of fundamental algorithm design but also underscored the importance of writing clear, well-organized code. Overall, it was a valuable learning experience that enhanced our grasp of algorithmic concepts and their practical applications.

Page no.: ____/
Date: ____/____/____

DAA Assignment

* Pseudo code

Linear Search (arr[ ], Key)
// find index of Key in array by linear Search.
// input : Array of integers & Key.
// output : Index of number if found else return -1.

```
for ( i=0 to i=n-1 , i++)
    if (arr[ ] == Key)
        return i ; // index of Key

return -1 //not found.
```

Binary Search (arr[ ], Key)
// find index of Key by Binary Search.
// input : array of int. that is sorted in acending order
// Output : index of Key if found else return -1.

Start = 0, end = size -1, mid = $\left(\frac{\text{Start+1 end}}{2}\right)$

```
While ( Start <= end )
{  if (arr [ mid ] == Key )
       return mid // index of Key
   if (arr [mid] < key)
       start = mid+1
   else
       end = mid-1;
}
return -1
```

Page no.: ____/
Date: ____/____/____/

Test - Case : Linear Search

	Input	Key	Output
1)	{1,2,3,4,5}	4	3
2)	{1,7,9,16,12}	9	2
3)	{1,3,15,16,80}	16	3
4)	{1,2,5,7,9}	6	-1
5)	{3,5,7,11}	15	-1
6)	{ }	4	-1

Test Case : Binary - Search

	Input	Key	Output
1)	{7,9,11,15,20}	11	2
2)	{2,4,6,8,10,12,14}	14	6
3)	{1,3,5,7,9,11,13}	11	5
4)	{20,40,60,80,100}	65	-1
5)	{1}	6	-1

Page no.: ____/

Date: ____/____/____/

Time Complexity :

(1) Linear Search.

(a) Best Case : // Key Present at first element

```
1  <-----  Cin << "enter target.";
1  <-----  for loop
1  <-----  Return Statement
```

$O(3) \simeq O(1)$

(b) Worst Case : // Key Present at end Position

```
1  <-----  Cin << "Enter target";
n  <-----  for loop
1  <-----  Return Statement.
```

$O(n+2) \simeq O(n)$.

(c) Avg. Case : // Key may be present in between position.

Let the Element to be present at the K^{th} position.

```
1  <-----  Cin << "Enter target";
K  <-----  for loop ( Run for Kth time)
1  <-----  Return Statement
```

$O(K+1) \simeq O(K)$

Page no.: ____/
Date: ____/____/____

(2) Binary Search

(a) Best Case: // Key Present at the mid position.

1 ⟵ Cin << "Enter the target";
1 ⟵ for loop
1 ⟵ return Statement.

$O(3) \approx O(1)$.

(b) Worst Case: // Key may not be present or present at any end of the array.

1 ⟵ Cin << "Enter the target";
log(n) ⟵ for loop.
⟶ because every time the number of element in the array becomes half.

(c) Avg Case: // Key at Random location.

Element present in between position but not at the center of the array. So the loop will run for K^{th} time.

1 ⟵ Cin "Enter target";
K ⟵ for loop (it runs for k times)
1 ⟵ Return Statement

$O(k+2) = O(k)$.