## Ex:11.A. IMPLEMENTATION OF SEARCHING ALGORITHMS LINEAR SEARCH AND BINARY SEARCH

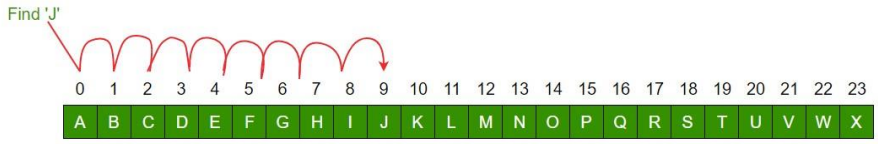
**AIM:**

To write a C Program to implement different searching techniques – Linear and Binary search.

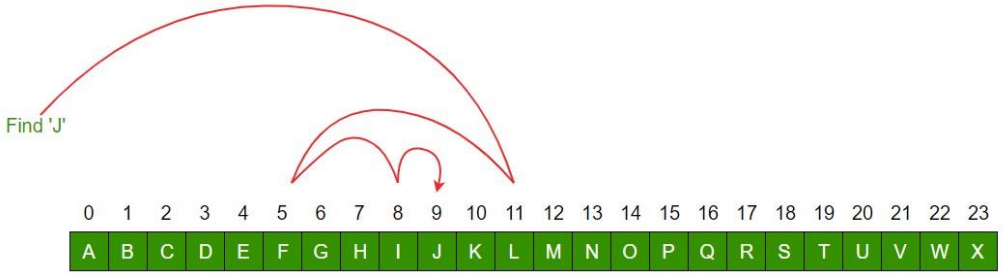
**DESCRIPTION:**

Binary search however, cut down your search to half as soon as you find middle of a sorted list.The middle element is looked to check if it is greater than or less than the value to be searched.Accordingly, search is done to either half of the given list

**Linear Search to find the element "J" in a given sorted list from A-X**




**Binary Search to find the element "J" in a given sorted list from A-X**




**ALGORITHM:**

**Linear Search**:

1. Read the search element from the user
2. Compare, the search element with the first element in the list.
3. If both are matching, then display "Given element found!!!" and terminate the function
4. If both are not matching, then compare search element with the next element in the list.
5. Repeat steps 3 and 4 until the search element is compared with the last element in the list.
6. If the last element in the list is also doesn't match, then display "Element not found!!!" and terminate the function.

**Binary search is implemented using following steps...**

1. Read the search element from the user
2. Find the middle element in the sorted list
3. Compare, the search element with the middle element in the sorted list.

4. If both are matching, then display "Given element found!!!" and terminate the function
5. If both are not matching, then check whether the search element is smaller or larger than middle element.
6. If the search element is smaller than middle element, then repeat steps 2, 3, 4 and 5 for the left sublist of the middle element.
7. If the search element is larger than middle element, then repeat steps 2, 3, 4 and 5 for the right sublist of the middle element.
8. Repeat the same process until we find the search element in the list or until sublist contains only one element.
9. If that element also doesn't match with the search element, then display "Element not found in the list!!!" and terminate the function.

**PROGRAM**

```
#include <stdio.h>
void sequential_search(int array[], int size, int n)
{
int
i;
    for (i = 0; i < size; i++)
    {
        if (array[i] == n)
        {
            printf("%d found at location %d.\n", n, i+1);
break;
        }
}     if (i
== size)
        printf("Not found! %d is not present in the list.\n",
n);
}
void binary_search(int array[], int size, int n)
{
```

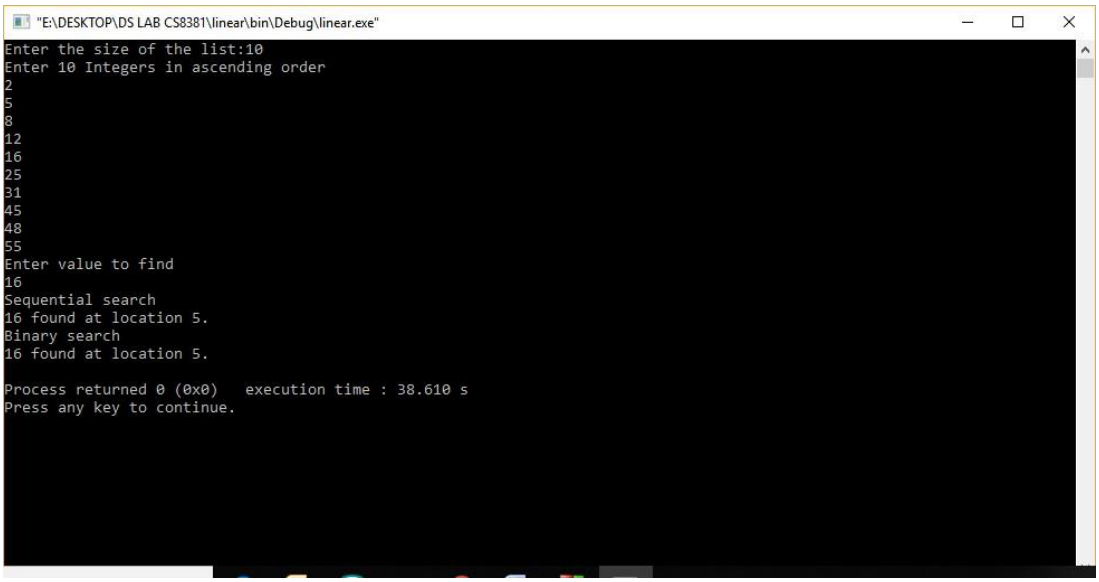
```
    int i, first, last,
middle;     first = 0;
last = size - 1;
    middle = (first+last) / 2;
     while (first <= last) {
if (array[middle] < n)
first = middle + 1;         else
if (array[middle] == n) {
            printf("%d found at location %d.\n", n,
middle+1);            break;         }         else
            last = middle - 1;

        middle = (first + last) / 2;
    }
    if ( first > last )
        printf("Not found! %d is not present in the list.\n",
n);
```

```
}

int main()
{
    int a[200], i, j, n, size;
    printf("Enter the size of the list:");
    scanf("%d", &size);
    printf("Enter %d Integers in ascending order\n", size);
    for (i = 0; i < size; i++)
        scanf("%d", &a[i]);
    printf("Enter value to find\n");
    scanf("%d", &n);
    printf("Sequential search\n");
    sequential_search(a, size, n);
    printf("Binary search\n");
    binary_search(a, size, n);
    return 0;
}
```

**OUTPUT**




**RESULT**

Thus the C Program to implement different searching techniques Linear and Binary search