



**Experiment No.: 08**

**Title: Implementation of searching algorithm**



Batch: B1

Roll No.: 16010420133

Experiment no.: 8

**Aim: To implement a binary search algorithm using an array.**

---

**Resources Used:** Turbo C/ C++ editor and C compiler.

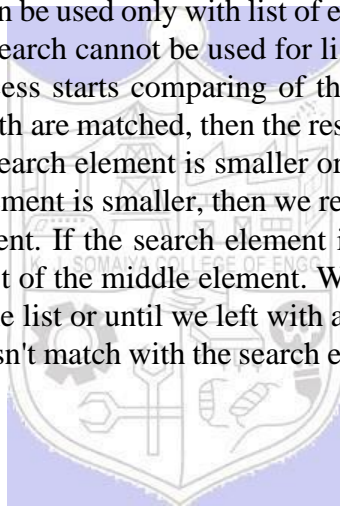
---

### Theory:

#### Searching –

Search is a process of finding a value in a list of values. In other words, searching is the process of locating given value position in a list of values.

The **binary search algorithm** can be used with only sorted list of element. That means, binary search can be used only with list of element which are already arranged in a order. The binary search cannot be used for list of element which are in random order. This search process starts comparing of the search element with the middle element in the list. If both are matched, then the result is "element found". Otherwise, we check whether the search element is smaller or larger than the middle element in the list. If the search element is smaller, then we repeat the same process for left sub-list of the middle element. If the search element is larger, then we repeat the same process for right sub-list of the middle element. We repeat this process until we find the search element in the list or until we left with a sub-list of only one element. And if that element also doesn't match with the search element, then the result is "Element not found in the list".



### Algorithm:

- 1) **PreCondition** - Sort the given input array using counting sort.

```

COUNTING-SORT( $A, B, k$ )
1  for  $i \leftarrow 1$  to  $k$ 
2      do  $C[i] \leftarrow 0$ 
3  for  $j \leftarrow 1$  to  $\text{length}[A]$ 
4      do  $C[A[j]] \leftarrow C[A[j]] + 1$ 
5   $\triangleright C[i]$  now contains the number of elements equal to  $i$ .
6  for  $i \leftarrow 2$  to  $k$ 
7      do  $C[i] \leftarrow C[i] + C[i - 1]$ 
8   $\triangleright C[i]$  now contains the number of elements less than or equal to  $i$ .
9  for  $j \leftarrow \text{length}[A]$  downto 1
10     do  $B[C[A[j]]] \leftarrow A[j]$ 
11         $C[A[j]] \leftarrow C[A[j]] - 1$ 
  
```

- 2) **Binary Search algorithm** -

```

BinarySearch(list[], min, max, key)
if max < min then
    return false
else
    mid = (max+min) / 2
    if list[mid] > key then
        return BinarySearch(list[], min, mid-1, key)
    else if list[mid] < key then
        return BinarySearch(list[], mid+1, max, key)
    else
        return mid
    end if
end if

```

---

**Code:**

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
void counting_sort(int a[],int b[],int n,int max)
```

```
{
```

```
    int count[50]={0},i,j,loop;
```

```
    int m =0;
```

```
    for(i=0;i<n;++i)
```

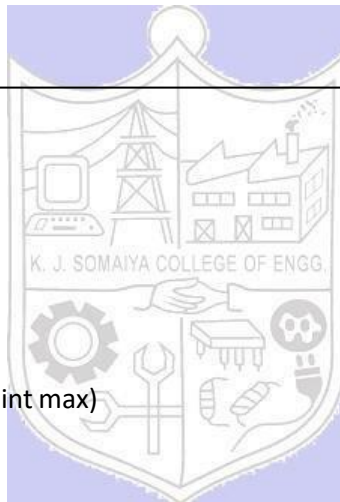
```
        count[a[i]]=count[a[i]]+1;
```

```
    printf("\nThe sorted elements are: ");
```

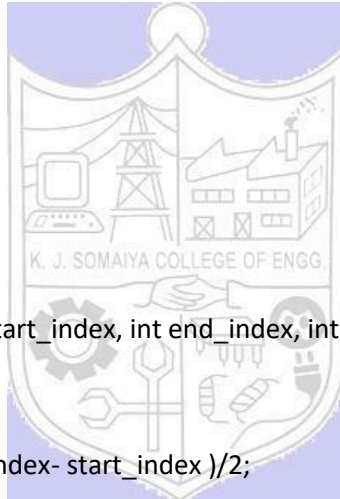
```
    for(i=0;i<=max;++i)
```

```
    {
```

```
        for(j=1;j<=count[i];++j)
```



```
{  
    // printf("%d",i);  
    while (m<=n)  
    {  
        b[m]= i;  
        m++;  
        break;  
    }  
}  
  
for(loop = 0; loop < n; loop++)  
    printf("%d ", b[loop]);  
}  
  
int iterativeBinarySearch(int b[], int start_index, int end_index, int element){  
    while (start_index <= end_index){  
        int middle = start_index + (end_index- start_index )/2;  
        if (b[middle] == element)  
            return middle;  
        if (b[middle] < element)  
            start_index = middle + 1;  
        else  
            end_index = middle - 1;  
    }  
    return -1;  
}
```



```
int main()
{
    int a[50],b[50],n,i,element,max=0;

    printf("Enter the number of elements: ");

    scanf("%d",&n);

    printf("\nEnter the elements: ");

    for(i=0;i<n;++i)
    {
        scanf("%d",&a[i]);

        if(a[i]>max)
            max=a[i];
    }

    counting_sort(a,b,n,max);

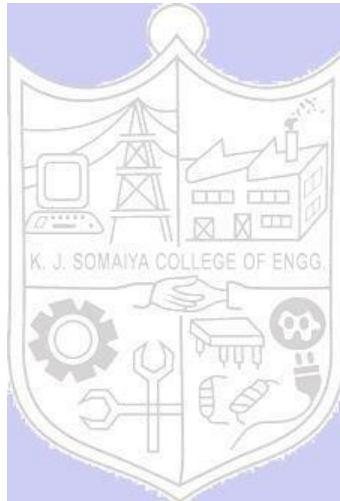
    printf("\nEnter the element to search: ");

    scanf("%d",&element);

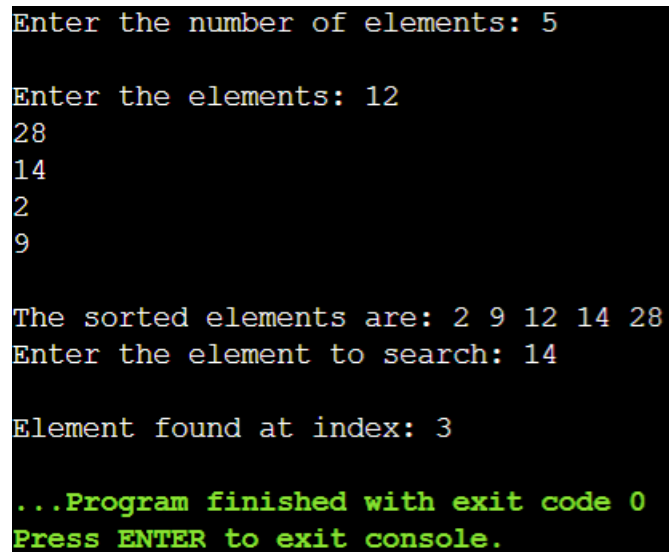
    int found_index = iterativeBinarySearch(b, 0, n-1, element);

    if(found_index == -1 )
    {
        printf("\nElement not found.");
    }

    else
```



```
{  
    printf("\nElement found at index: %d",found_index);  
}  
  
return 0;  
}
```

**Output:**

```
Enter the number of elements: 5  
  
Enter the elements: 12  
28  
14  
2  
9  
  
The sorted elements are: 2 9 12 14 28  
Enter the element to search: 14  
  
Element found at index: 3  
  
...Program finished with exit code 0  
Press ENTER to exit console.
```

**Outcomes:**

CO4: Demonstrate different sorting and searching methods.

**Conclusion:**

Implemented binary search algorithm using array understood the concept.

**References:****Books/ Journals/ Websites:**

- Y. Langsam, M. Augenstein and A. Tenenbaum, “Data Structures using C”, Pearson Education Asia, 1st Edition, 2002.
- Vlabs on binary search and counting sort.