University of Information Technology and Sciences (UITS)

Department of Information Technology

LAB-1:

IT-214 : Algorithm Lab

Searching Algorithm

Submitted To:

Sumaiya Akhtar Mitu
Lecturer,
Department of IT, UITS

Submitted By:

Name:Nazmul Zaman
Student ID:2014755055
Department of IT, UITS

April 9, 2022

Contents

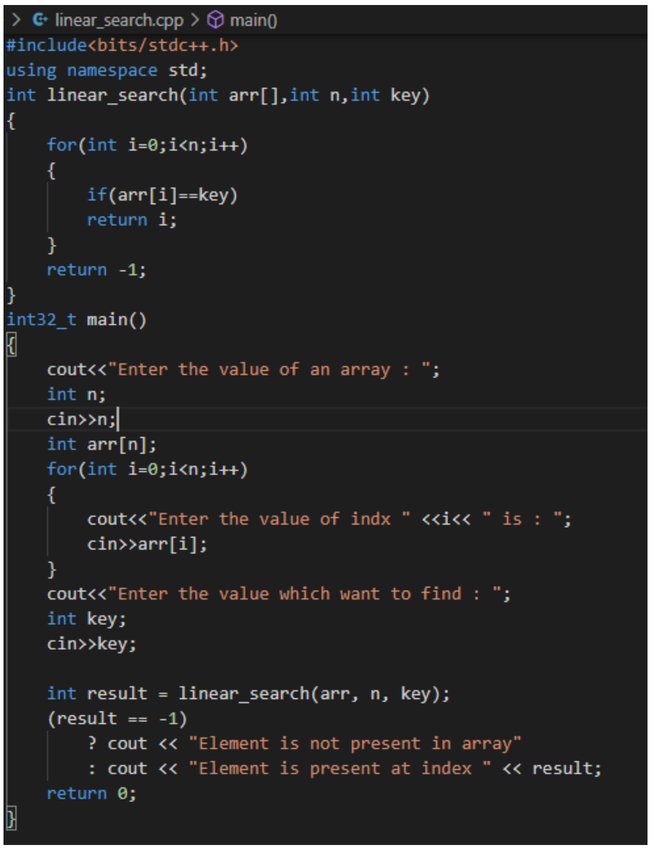
1 Abstraction

In this lab report we learn how to worked searching algorithm and use of searching algorithm and why we use searching algorithm and best way for use . 1.Linear Search 2.Binary Search These two type searching algorithm help us to find the element is present in array or not and which is useful to use and best way to implement in program.

2 Introduction

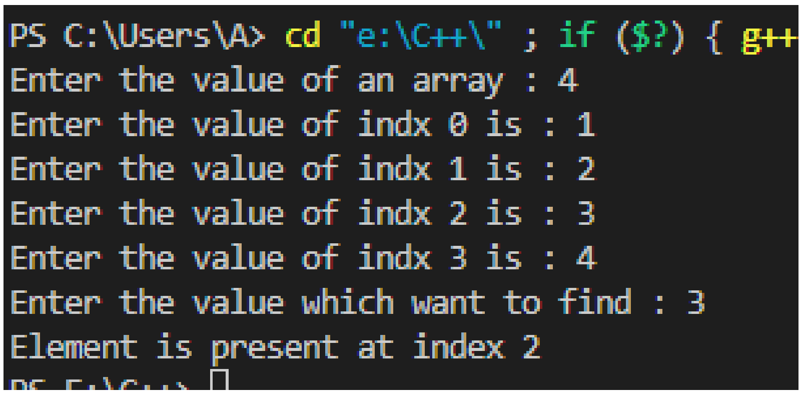
Linear search is a search that finds an element in the list by searching the element sequentially until the element is found in the list. On the other hand, a binary search is a search that finds the middle element in the list recursively until the middle element is matched with a searched element.

3 Source code and Output



```cpp
#include<bits/stdc++.h>
using namespace std;
int linear_search(int arr[],int n,int key)
{
    for(int i=0;i<n;i++)
    {
        if(arr[i]==key)
        return i;
    }
    return -1;
}
int32_t main()
{
    cout<<"Enter the value of an array : ";
    int n;
    cin>>n;
    int arr[n];
    for(int i=0;i<n;i++)
    {
        cout<<"Enter the value of indx " <<i<< " is : ";
        cin>>arr[i];
    }
    cout<<"Enter the value which want to find : ";
    int key;
    cin>>key;

    int result = linear_search(arr, n, key);
    (result == -1)
        ? cout << "Element is not present in array"
        : cout << "Element is present at index " << result;
    return 0;
}
```

Figure 1: Linear Search



```
PS C:\Users\A> cd "e:\C++\" ; if ($?) { g++
Enter the value of an array : 4
Enter the value of indx 0 is : 1
Enter the value of indx 1 is : 2
Enter the value of indx 2 is : 3
Enter the value of indx 3 is : 4
Enter the value which want to find : 3
Element is present at index 2
```

Figure 2: Output

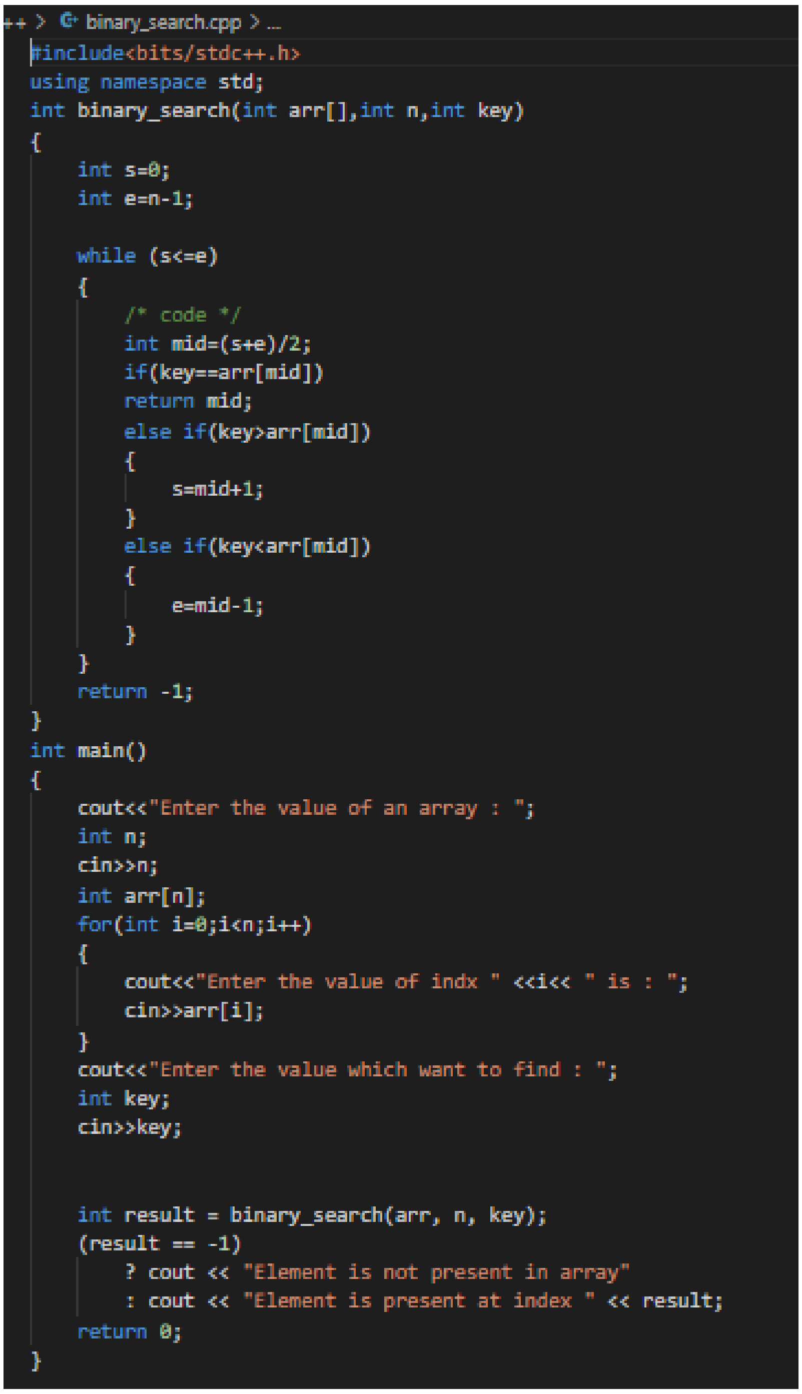
4 Conclusion

In this lab report we learn about different type of searching algorithm and use of searching algorithm . Search algorithms work to retrieve information stored within some data structure, or calculated in the search space of a problem domain, with either discrete or continuous values. By using these algorithm we can solve different type of searching algorithm problem and it's also helpful for competitive programming.

5 References

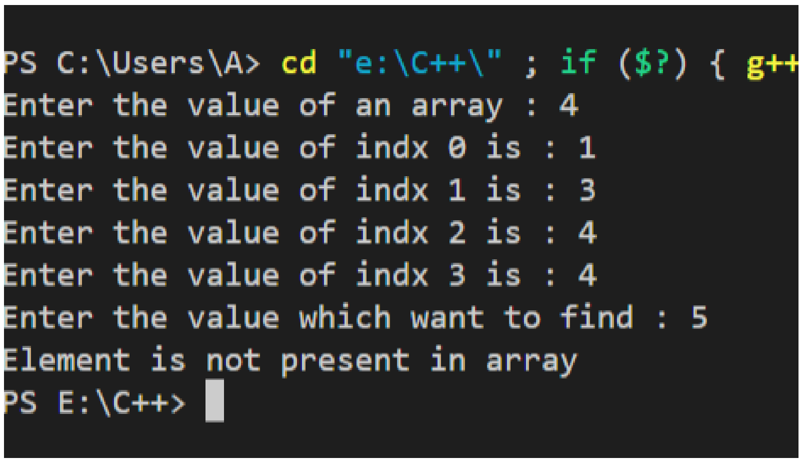
References

1.https://github.com/ZamanNazmul/Data$_{s}tructure$



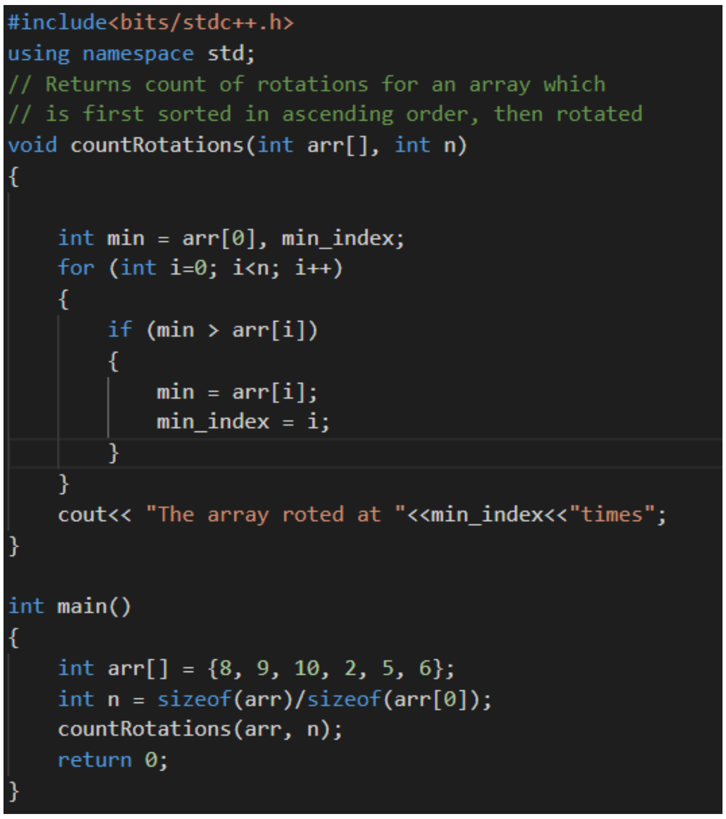
```cpp
#include<bits/stdc++.h>
using namespace std;
int binary_search(int arr[],int n,int key)
{
    int s=0;
    int e=n-1;

    while (s<=e)
    {
        /* code */
        int mid=(s+e)/2;
        if(key==arr[mid])
        return mid;
        else if(key>arr[mid])
        {
            s=mid+1;
        }
        else if(key<arr[mid])
        {
            e=mid-1;
        }
    }
    return -1;
}
int main()
{
    cout<<"Enter the value of an array : ";
    int n;
    cin>>n;
    int arr[n];
    for(int i=0;i<n;i++)
    {
        cout<<"Enter the value of indx " <<i<< " is : ";
        cin>>arr[i];
    }
    cout<<"Enter the value which want to find : ";
    int key;
    cin>>key;


    int result = binary_search(arr, n, key);
    (result == -1)
        ? cout << "Element is not present in array"
        : cout << "Element is present at index " << result;
    return 0;
}
```

Figure 3: Binary Search



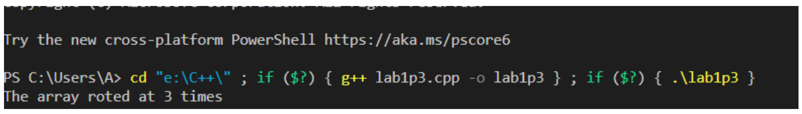
```
PS C:\Users\A> cd "e:\C++\" ; if ($?) { g++
Enter the value of an array : 4
Enter the value of indx 0 is : 1
Enter the value of indx 1 is : 3
Enter the value of indx 2 is : 4
Enter the value of indx 3 is : 4
Enter the value which want to find : 5
Element is not present in array
PS E:\C++>
```

Figure 4: Output



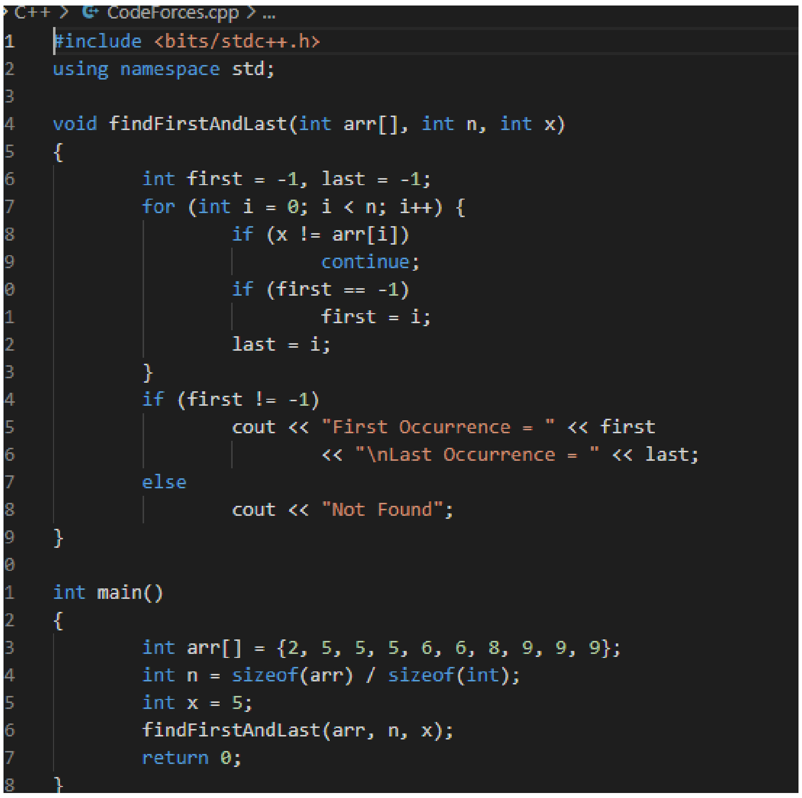
```cpp
#include<bits/stdc++.h>
using namespace std;
// Returns count of rotations for an array which
// is first sorted in ascending order, then rotated
void countRotations(int arr[], int n)
{

    int min = arr[0], min_index;
    for (int i=0; i<n; i++)
    {
        if (min > arr[i])
        {
            min = arr[i];
            min_index = i;
        }
    }
    cout<< "The array roted at "<<min_index<<"times";
}

int main()
{
    int arr[] = {8, 9, 10, 2, 5, 6};
    int n = sizeof(arr)/sizeof(arr[0]);
    countRotations(arr, n);
    return 0;
}
```

Figure 5: Rotated Count



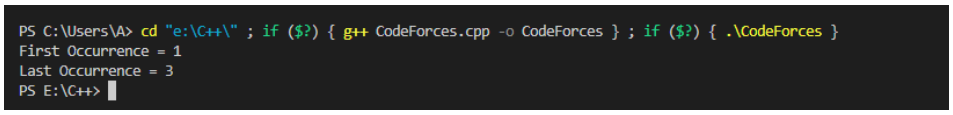
```
Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\A> cd "e:\C++\" ; if ($?) { g++ lab1p3.cpp -o lab1p3 } ; if ($?) { .\lab1p3 }
The array roted at 3 times
```

Figure 6: Output



```
C++ > CodeForces.cpp > ...
#include <bits/stdc++.h>
using namespace std;

void findFirstAndLast(int arr[], int n, int x)
{
        int first = -1, last = -1;
        for (int i = 0; i < n; i++) {
                if (x != arr[i])
                        continue;
                if (first == -1)
                        first = i;
                last = i;
        }
        if (first != -1)
                cout << "First Occurrence = " << first
                        << "\nLast Occurrence = " << last;
        else
                cout << "Not Found";
}

int main()
{
        int arr[] = {2, 5, 5, 5, 6, 6, 8, 9, 9, 9};
        int n = sizeof(arr) / sizeof(int);
        int x = 5;
        findFirstAndLast(arr, n, x);
        return 0;
}
```

Figure 7: Output



```
PS C:\Users\A> cd "e:\C++\" ; if ($?) { g++ CodeForces.cpp -o CodeForces } ; if ($?) { .\CodeForces }
First Occurrence = 1
Last Occurrence = 3
PS E:\C++>
```

Figure 8: Output