DAA LAB ASSIGNMENT 1

Name: Aditya Anand

Roll No.: 20124009

Branch: IT

S No.	*Title*	*Date*	*Remarks*
1	Sequential and Binary Search	06-01-2022	
2	Selection Sort	06-01-2022	

Sequential And Binary Search

INPUT: Given array of n integers in sorted order and a target to be searched

OUTPUT: True or False, based on whether target is present in the array or not

TIME COMPLEXITY:

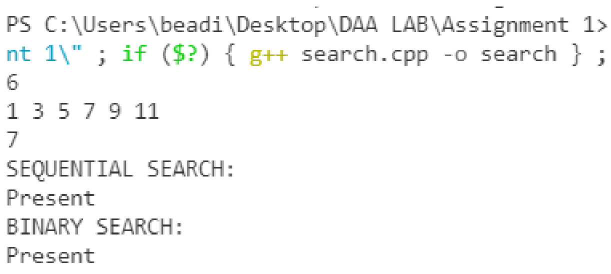
O(n) (For sequential search)

O(log(n)) (For binary search)

CODE:

```
#include<bits/stdc++.h>
using namespace :: std;

// O(n)
bool seq_search(vector<int> v, int target){
    for(int i=0; i<v.size(); i++){
        if(target==v[i]){
            return true;
        }
    }
    return false;
}

// O(log(n))
bool binarySearch(vector<int> v, int target){
    int i=0, j=v.size()-1;
    while(j>=i){
        int mid = i + (j-i)/2;
        if(v[mid]==target){
            return true;
        }
        else if(v[mid]>target){
            j=mid-1;
        }
        else{
            i=mid+1;
        }
    }
    return false;
}

int main(){
    int n = 0;
    cin>>n;
    vector<int> v(n, 0);
    for(int i=0; i<n; i++){
        cin>>v[i];
    }

    int target = 0;
    cin>>target;
    cout<<"SEQUENTIAL SEARCH:\n";
    if(seq_search(v, target)){
        cout<<"Present\n";
    }
    else{
        cout<<"Not Present\n";
    }
    cout<<"BINARY SEARCH:\n";
    if(binarySearch(v, target)){
        cout<<"Present\n";
    }
    else{
        cout<<"Not Present\n";
    }

    return 0;
}
```

RESULT:



```
PS C:\Users\beadi\Desktop\DAA LAB\Assignment 1>
nt 1\" ; if ($?) { g++ search.cpp -o search } ;
6
1 3 5 7 9 11
7
SEQUENTIAL SEARCH:
Present
BINARY SEARCH:
Present
```

Selection Sort

INPUT: an array of n integers

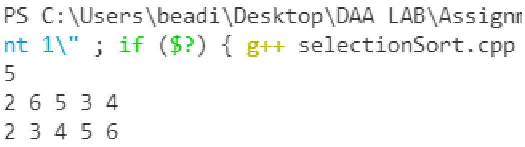
OUTPUT: sorted array

TIME COMPLEXITY: O(n^2)

CODE:

```
#include<bits/stdc++.h>
using namespace :: std;

// O(n^2)
void selection_sort(vector<int> &v){
    // Array from 0 to i-1 is sorted, i to n is unsorted
    int i=0;
    while(i<v.size()){
        int mn = INT_MAX;
        int id = -1;
        for(int j=i; j<v.size(); j++){
            if(mn>v[j]){
                mn = v[j];
                id = j;
            }
        }
        swap(v[i], v[id]);
        i++;
    }
}

int main(){
    int n=0;
    cin>>n;
    vector<int> v(n, 0);
    for(int i=0; i<n; i++){
        cin>>v[i];
    }
    selection_sort(v);
    for(auto i:v){
        cout<<i<<" ";
    }
    return 0;
}
```

RESULT:



```
PS C:\Users\beadi\Desktop\DAA LAB\Assignm
nt 1\" ; if ($?) { g++ selectionSort.cpp
5
2 6 5 3 4
2 3 4 5 6
```

TIME COMPLEXITY ANALYSIS FOR SELECTION SORT:

Since we use to nested loops from 0 to n-1, one for increasing the partition of sorted and unsorted arrays and the other to find the min in the unsorted array, the time complexity of the above algorithm is quadratic i.e. O(n^2)