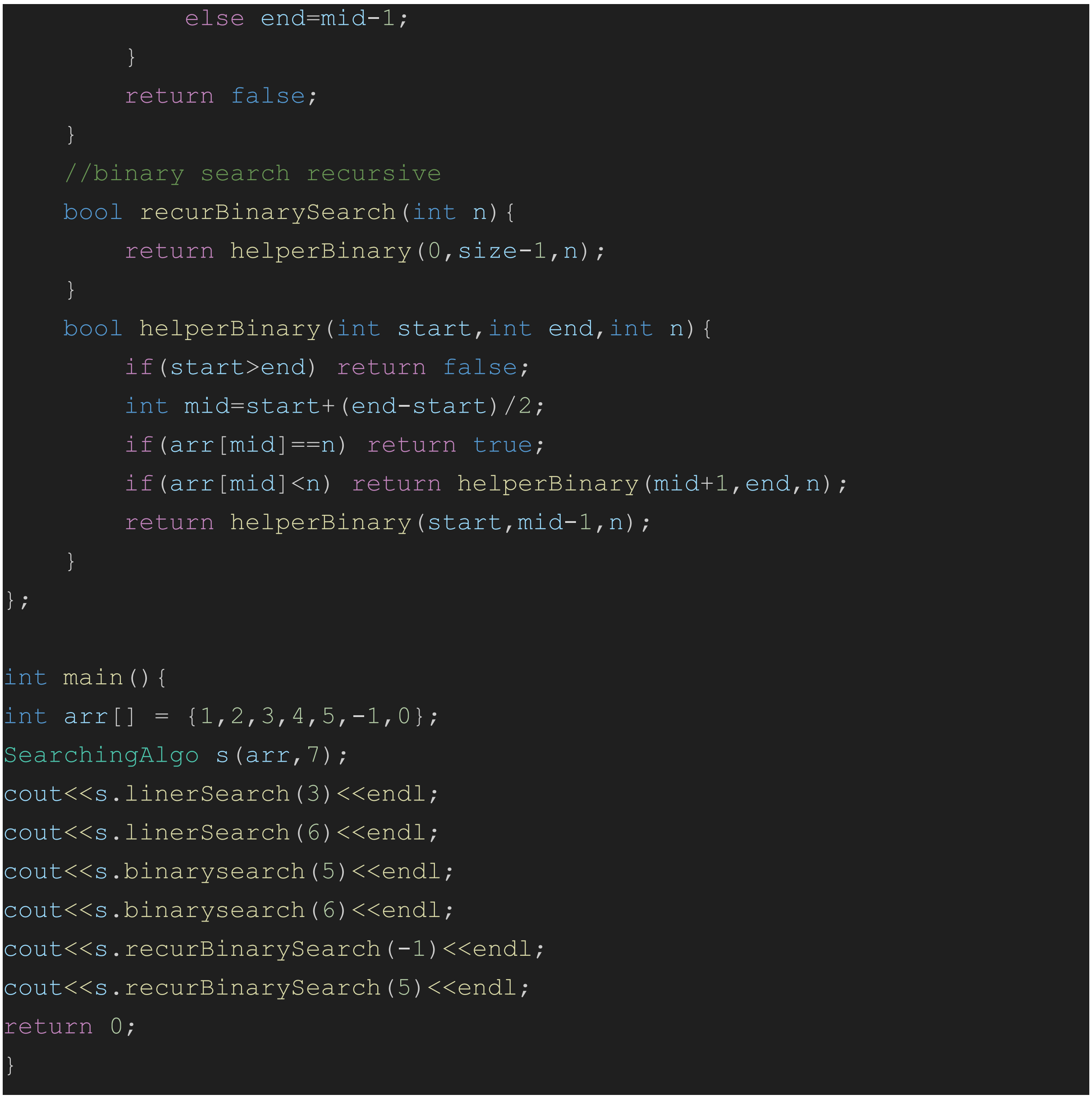
Name : Vikrant   Roll:64   Assignment : Implementation of Searching Algos

```
#include<iostream>
#include<algorithm>
using namespace std;
class SearchingAlgo{
    int size;
    int *arr;
    public:
    //constructor
    SearchingAlgo(int arr[],int size){
        sort(arr,arr+size);
        for(int i=0;i<size;i++){
            cout<<arr[i]<<" ";
        }
        this->size=size;
        this->arr=new int[size];
        for(int i=0;i<size;i++){
            this->arr[i]=arr[i];
        }
    }
    //question 1
    //linear search returning true or false
    bool linerSearch(int n){
        for(int i=0;i<size;i++){
            if(arr[i]==n) return true;
        }
        return false;
    }
    //linear search recursive
    bool recursiveLinear(int n){
        return helperLinear(0,n);
    }
    bool helperLinear(int idx,int n){
        if(idx==size) return false;
        return arr[idx]==n ||helperLinear(idx+1,n);
    }
    //binarysearch iterative
    bool binarysearch(int n){
        int start=0;
        int end =size-1;
        while(start<=end){
            int mid=start+(end-start)/2;
            if(arr[mid]==n) return true;
            if(arr[mid]<n) start=mid+1;
```



```cpp
            else end=mid-1;
        }
        return false;
    }
    //binary search recursive
    bool recurBinarySearch(int n){
        return helperBinary(0,size-1,n);
    }
    bool helperBinary(int start,int end,int n){
        if(start>end) return false;
        int mid=start+(end-start)/2;
        if(arr[mid]==n) return true;
        if(arr[mid]<n) return helperBinary(mid+1,end,n);
        return helperBinary(start,mid-1,n);
    }
};

int main(){
int arr[] = {1,2,3,4,5,-1,0};
SearchingAlgo s(arr,7);
cout<<s.linerSearch(3)<<endl;
cout<<s.linerSearch(6)<<endl;
cout<<s.binarysearch(5)<<endl;
cout<<s.binarysearch(6)<<endl;
cout<<s.recurBinarySearch(-1)<<endl;
cout<<s.recurBinarySearch(5)<<endl;
return 0;
}
```