Divide&conquer

1

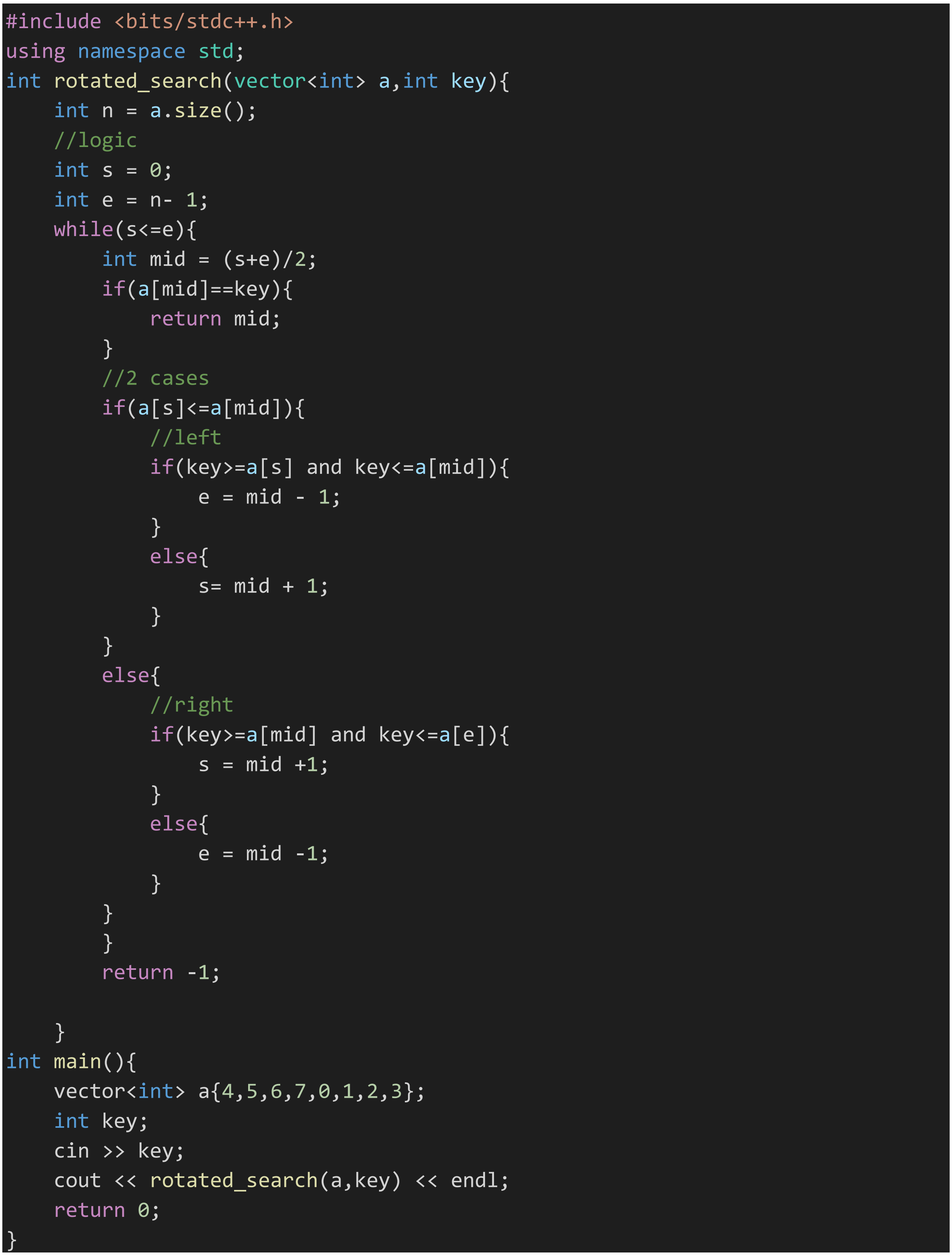
```
//merge sort
#include <bits/stdc++.h>
using namespace std;
void merge(vector<int> &arr,int s,int e){
    int i = s;
    int m = (s+e)/2;
    int j = m + 1;
    vector<int> temp;
    while(i<=m and j<=e){
        if(array[i] < array[j]){
            temp.push_back(array[i]);
            i++;
        }
        else{
            temp.push_back(array[j]);
            j++;
        }
    }
    //this loop

}
void merge_sort(vector<int> &arr , int s , int e){
        //base case
    if(s>=e){
        return;
    }
    //rec case
    int mid = (s+e)/2;
    merge_sort(arr,s,mid);
    merge_sort(arr,mid+1,e);
    return merge(arr,s,e);

}
int main(){
    vector<int> arr{5,3,4,643,24,6,85,35};
    int s = 0;
    int e = arr.size()-1;
    merge_sort(arr,s,e);
    for(int x : arr)
{
    cout << x << " ";
}

}
```

2



```
#include <bits/stdc++.h>
using namespace std;
int rotated_search(vector<int> a,int key){
    int n = a.size();
    //logic
    int s = 0;
    int e = n- 1;
    while(s<=e){
        int mid = (s+e)/2;
        if(a[mid]==key){
            return mid;
        }
        //2 cases
        if(a[s]<=a[mid]){
            //left
            if(key>=a[s] and key<=a[mid]){
                e = mid - 1;
            }
            else{
                s= mid + 1;
            }
        }
        else{
            //right
            if(key>=a[mid] and key<=a[e]){
                s = mid +1;
            }
            else{
                e = mid -1;
            }
        }
        }
        return -1;

    }
int main(){
    vector<int> a{4,5,6,7,0,1,2,3};
    int key;
    cin >> key;
    cout << rotated_search(a,key) << endl;
    return 0;
}
```