mo.cpp

```cpp
//Given a sequence of n numbers a1, a2, ..., an and a number of d-queries.
//A d-query is a pair (i, j) (1 ≤ i ≤ j ≤ n). For each d-query (i, j), you have
//to return the number of distinct elements in the subsequence ai, ai+1, ..., aj.
#include<bits/stdc++.h>
#define TAM 30000 + 7
#define QTAM 200000 + 7
#define MTAM 1000000 + 7

using namespace std;

int a[TAM], r[QTAM], cnt[MTAM];
int ans, BLOCK, currL , currR ;

struct node{
    int L, R, idx;
}q[QTAM];

bool comp(node a, node b){
    if(a.L/BLOCK < b.L/BLOCK) return true;
    if(a.L/BLOCK > b.L/BLOCK) return false;
    return a.R < b.R;
}

void remove(int i){
    cnt[a[i]]--;
    if(cnt[a[i]] == 0)ans--;
}

void add(int i){
    cnt[a[i]]++;
    if(cnt[a[i]]==1)ans++;
}

int query(node i){
    while(currL< i.L){
        remove(currL);
        currL++;
    }
    while(currL > i.L){
        currL--;
        add(currL);
    }
    while(currR< i.R){
        currR++;
        add(currR);
    }
    while(currR > i.R){
        remove(currR);
        currR--;
    }
    return ans;
}

int main(){
    int n, que;
    cin>>n;
    BLOCK = sqrt(n);
    for(int i = 1; i <= n; i++)cin>>a[i];

    cin>>que;
    for(int i = 1; i <= que; i++){
```

```
        cin>>q[i].L>>q[i].R;
        q[i].idx = i;
    }
    sort(q +1, q + que+1, comp);

    for(int i = 1; i <=que; i++)
        r[q[i].idx] = query(q[i]);

    for(int i = 1; i <= que; i++)
        cout<<r[i]<<"\n";
}
```