```cpp
/*** Lightoj - 1188: Fast Queries
Given an array of N integers indexed from 1 to N, and q queries, each in the form i j,
you have to find the number of distinct integers from index i to j (inclusive).
***/

#include<bits/stdc++.h>
using namespace std;
#define mx 100005
int n,q,a[mx],cnt[mx],last[mx],tree[mx];
struct dt{ int xx,yy,id,rs; }st[mx];
bool cmp1(dt pp, dt qq){ return pp.yy<qq.yy; }
bool cmp2(dt pp, dt qq){ return pp.id<qq.id; }

void update(int idx,int v){
    while(idx<=n){
        tree[idx] += v;
        idx += (idx) & (-idx);
    }
}
int query(int idx){
    int sum = 0;
    while(idx>0){
        sum += tree[idx];
        idx -= (idx) & (-idx);
    }
    return sum;
}
int main(){
    int tt; scanf("%d",&tt);
    for(int ks=1; ks<=tt; ks++){
        printf("Case %d:\n",ks);
        scanf("%d%d",&n,&q);

        for(int i=1; i<=n; i++) scanf("%d",&a[i]);
        for(int i=1; i<=n; i++) cnt[i] = 0;

        for(int i=1; i<=q; i++){
            int xx,yy; scanf("%d%d",&xx,&yy);
            st[i].xx = xx; st[i].yy = yy;
            st[i].id = i;  st[i].rs = 0;
            cnt[yy]++;
        }
        // sort ascending order in term of upper bound
        sort(st+1,st+q+1,cmp1);

        for(int i=0; i<mx; i++) last[i] = 0;
        for(int i=1; i<=n; i++) tree[i] = 0;
        int k=1;

        for(int i=1; i<=n; i++){
            int v = a[i];
            if(last[v]!=0) update(last[v],-1);

            update(i,1);
            last[v]=i;
            for(int j=1; j<=cnt[i]; j++){
                int xx = st[k].xx; int yy = st[k].yy;
                int rs = query(yy)-query(xx-1);
                st[k].rs = rs;
                k++;
            }
        }
        // sort ascending order in term of id in query
        sort(st+1,st+q+1,cmp2);
        for(int i=1; i<=q; i++) printf("%d\n",st[i].rs);
    }
    return 0;
}
```