Fast Quries(Offline Segment tree)

Prob: how many distinct Character in a given interval

Idea:

I have traverse from beginning to end and updated the immediate last position of the charcter with (0) and this position with (1)

```
#include<bits/stdc++.h>
using namespace std;
#define ll long long
struct st
{
   ll frm,to,val,id;
} Q[100005];

ll tree[100005 * 4];

ll prev[100005];
ll Val[100005];
ll arr[100005];
bool comp(st A, st B)
{
   if(A.to == B.to)
      return A.frm<B.frm;
   else return A.to<B.to;
}
bool comp1(st A, st B)
{
      return A.id<B.id;
}

void update(long long node, long long b, long long e, long long i,long long v)
{
   if(e<i || b>i) return;
   if(b== e)
   {
      if(v== 0)
      {
         tree[node]-=1;
         Val[i] = 0;
      }
      else
      {
         tree[node]+=1;
         Val[i] = 1;
      }
      return;
   }
   ll left = node * 2;

   ll right = node * 2 + 1;

   ll mid = (b + e)/2;

   update(left,b , mid,i,v);

   update(right,mid + 1 , e,i,v);

   tree[node] = tree[left] + tree[right];

}

ll query(long long node, long long b, long long e, long long i, long long j)
{
   if(e<i || b>j) return 0 ;
   if(b>= i &&  e<=j)
   {
      return tree[node];
   }
   ll left = node * 2;
```

```
    ll right = node * 2 + 1;

    ll mid = (b + e)/2;

    ll x = query(left,b , mid,i,j);

    ll y = query(right,mid + 1 , e,i,j);

    return x + y;

}
int main()
{
    ll t,w = 0;
    scanf("%lld",&t);
    while(t--)
    {
        memset(prev,-1,sizeof(prev));
        memset(Val,0,sizeof(Val));
        memset(tree,0,sizeof(tree));
        ll n,q;
        scanf("%lld %lld",&n,&q);
        for(int i=1; i<=n; i++)
            scanf("%lld",&arr[i]);
        for(int i=0; i<q; i++)
        {
            scanf("%lld %lld",&Q[i].frm,&Q[i].to);
            Q[i].id = i;
        }
        int track = 0;
        sort(Q,Q+q,comp);
        printf("Case %lld:\n",++w);
        for(int i=1; i<=n; i++)
        {
            if(prev[arr[i]] == -1)
            {
                update(1,1,n,i,1);
                prev[arr[i]] = i;
            }
            else
            {
                update(1,1,n,prev[arr[i]],0);
                update(1,1,n,i,1);
                prev[arr[i]] = i;
            }
            while(track<q && Q[track].to == i)
            {
                ll ans = query(1,1,n,Q[track].frm,Q[track].to);
                Q[track].val = ans;
                track++;
            }
        }
        sort(Q,Q+q,comp1);
        for(int i=0; i<q; i++)
        {
    printf("%lld\n",Q[i].val);
        }
    }
    return 0;
}
```