```cpp
/*** Sherlock and Inversions
Tag: Data Structure, MO's Algo, BIT, Segment Tree
Given an array of N integers denoted by A1,A2...AN and Q queries of form Li, Ri.
For each such you have to report the number of inversions in subarray denoted by [Li, Ri].
Inversions in a subarray denoted by [a, b] are number of pairs (i,j) such that
a ≤ i < j ≤ b and Ai > Aj. 1 ≤ N, Q ≤ 105; 1 ≤ Ai ≤ 109; 1 ≤ Li ≤ Ri ≤ N.
***/
#include<bits/stdc++.h>
using namespace std;
#define mx 100005
#define ll long long
const int BLOCK_SIZE = 320;
int N,aa[mx],bb[mx],tree[mx];
ll ans[mx],sum;
unordered_map<int,int>mp;
struct dt{ int l,r,x,b;}qr[mx];
bool cmp(dt p,dt q){
    if(p.b==q.b)return p.r<q.r;
    return p.b<q.b;
}
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 n,q; scanf("%d%d",&n,&q);
    for(int i=1; i<=n; i++){
        scanf("%d",&aa[i]); bb[i]=aa[i];
    }
    sort(bb+1,bb+n+1);
    int k=0;
    for(int i=1; i<=n; i++){
        if(mp.find(bb[i])==mp.end())mp[bb[i]]=++k;
    }
    N = k;
    for(int i=1; i<=n; i++) aa[i] = mp[aa[i]];
    for(int i=1; i<=q; i++){
        int l,r; scanf("%d%d",&l,&r);
        qr[i].l = l; qr[i].r = r; qr[i].x = i;
        qr[i].b = (l/BLOCK_SIZE)+1;
    }
    sort(qr+1,qr+q+1,cmp);
    int l=1,r=0; ll sum=0;
    for(int i=1; i<=q; i++){
        while(l<qr[i].l){
            sum -= query(aa[l]-1); update(aa[l],-1); l++;
        }
        while(l>qr[i].l){
            --l; sum += query(aa[l]-1); update(aa[l],1);
        }
        while(r<qr[i].r){
            ++r; sum += query(N)-query(aa[r]); update(aa[r],1);
        }
        while(r>qr[i].r){
            sum -= query(N)-query(aa[r]); update(aa[r],-1); r--;
        }
        ans[qr[i].x] = sum;
    }
    for(int i=1; i<=q; i++) printf("%lld\n",ans[i]);
    return 0;
}
```