```cpp
/********   segment lazy  ********/
#include <cstdio>
#include <iostream>
#include <cstdlib>
#include <cstring>
#include <vector>
#include <algorithm>
#include <map>

#define mx 160003
using namespace std;

struct vo
{
    int sum;
    int prop;
} tree[mx * 3];;
int n;
map<int,int>m;
vector<int>v;


int query(int node, int b, int e, int i, int j,int cr=0)
{
    if (i > e || j < b)
        return 0; //????? ??? ??????
    if (b >= i && e <= j)
        return tree[node].sum+cr*(e-b+1); //????????? ????????
    int Left = node * 2; //???? ????? ???
    int Right = node * 2 + 1;
    int mid = (b + e) / 2;
    int p1 = query(Left, b, mid, i, j,cr+tree[node].prop);
    int p2 = query(Right, mid + 1, e, i, j,cr+tree[node].prop);
    return p1 + p2; //??? ??? ??? ????? ??????
}
void update(int node, int b, int e, int i,int j, int newvalue)
{
    if (i > e || j < b)
        return; //?????? ??? ??????
    if (b >= i && e <= j) {
        //????????? ????????

tree[node].sum=tree[node].sum+newvalue;

                    tree[node].prop+=1;
        return;
    }

    int Left = node * 2; //???? ????? ???
    int Right = node * 2 + 1;
    int mid = (b + e) / 2;
    update(Left, b, mid, i, j, newvalue);
    update(Right, mid + 1, e, i, j, newvalue);
    tree[node].sum = (tree[Left].sum + tree[Right].sum+tree[node].prop);
}
int main()
{
    int a,b,c,d,e,f,i,j,T,p,q;
    scanf("%d",&T);
    for(i=0;i<T;i++)
```

```
60        {    m.clear();
61             memset(tree,0,sizeof(tree));
62             memset(arr,-1,sizeof(arr));
63             memset(bhol,0,sizeof(bhol));
64
65
66            scanf("%d %d",&a,&b);
67
68
69
70         for(j=1;j<=a;j++)
71         {
72             scanf("%d %d",&arr[j][0],&arr[j][1]);
73
74             v.push_back(arr[j][0]);
75             v.push_back(arr[j][1]);
76
77         }
78         for(j=1;j<=b;j++)
79         {
80           scanf("%d",&e);
81           bhol[j]=e;
82           v.push_back(e);
83
84         }
85
86         sort(v.begin(),v.end());
87         e=v.size();
88         p=1;
89         for(j=0;j<e;j++)
90         {   if(v[j]==v[j-1])
91                {
92                   m[v[j]]=m[v[j-1]];
93                }
94            else
95             {m[v[j]]=p;
96              p++;
97             }
98         }
99
100        for(j=1;j<=a;j++)
101        {
102
update(1,1,mx,m[arr[j][0]],m[arr[j][1]],1);
103        }
104        printf("Case %d:\n",i+1);
105
106
107        for(j=1;j<=b;j++)
108        {
109
110
printf("%d\n",query(1,1,mx,m[bhol[j]],m[bhol[j]]));
111        }
112
113       }
114
115       return 0;
116   }
```