







#include<bits/stdc++.h>

#define ll long long

#define inf 1e18

#define mn 2000005

using namespace std;

ll ans=0,n,m,bl,s[mn],pos[mn],c[mn],out[mn],

qn=0,tn=0;//qn，tn为询问和修改操作的个数

struct w{ll l,r,id,tim;}q[mn];//询问操作

struct ww{ll x,v;}b[mn];//修改操作

inline void in(ll &x)

{ll s=0,w=1;char ch=getchar();

while(ch<'0'||ch>'9'){if(ch=='-')w=-1;ch=getchar();}

while(ch>='0'&&ch<='9'){s=s\*10+ch-'0';ch=getchar();}

x=s\*w;

}

bool cmp(w a,w b)

{

if(pos[a.l]!=pos[b.l])return pos[a.l]<pos[b.l];

if(pos[a.r]!=pos[b.r])return pos[a.r]<pos[b.r];

return a.tim<b.tim;

}

void add(ll x){if(++s[x]==1)ans++;}//添加操作

void del(ll x){if(--s[x]==0)ans--;}//删除操作

void work(ll now,ll i)

{

if(b[now].x>=q[i].l&&b[now].x<=q[i].r)

{

if(--s[c[b[now].x]]==0)ans--;

if(++s[b[now].v]==1)ans++;

}

swap(b[now].v,c[b[now].x]);

//这里有个很巧妙的操作

//对于一个操作，下一次需要为的颜色是本次被改变的颜色

//比如，我把颜色3改为了7，那么再进行这次修改的时候就是把7改为3

//所以直接交换两种颜色就好，如果没有这样的性质，这需要重新写时间轴上的++和--操作

}

void solve()

{ll i,l=1,r=0,now=0;

for(i=1;i<=qn;i++)

{

while(l<q[i].l)del(c[l++]);//l小了就删

while(l>q[i].l)add(c[--l]);//l大了就加

while(r>q[i].r)del(c[r--]);//r大了就删

while(r<q[i].r)add(c[++r]);//r小了就加

while(now<q[i].tim)work(++now,i);//改少了，改过去

while(now>q[i].tim)work(now--,i);//改多了，改回来

out[q[i].id]=ans;//统计答案

}

for(i=1;i<=qn;i++)

printf("%lld\n",out[i]);

}

int main()

{

ll x,y,z,i,j,k;

char ch[10];

scanf("%lld%lld",&n,&m);

for(i=1;i<=n;i++)

in(c[i]);

for(i=1;i<=m;i++)

{

scanf("%s",ch);

if(ch[0]=='Q')

{

qn++;

in(q[qn].l);in(q[qn].r);

q[qn].tim=tn;//别忘了记录最近的修改位置

q[qn].id=qn;

}

else

{

tn++;

in(b[tn].x);in(b[tn].v);

}

}

bl=pow(n,2.0/3);//理论上的比较优的时间复杂度

for(i=1;i<=n;i++)

pos[i]=(i-1)/bl+1;

sort(q+1,q+qn+1,cmp);

solve();

return 0;

}