董炜隽

```
#include <cstdio>

#include <cctype>

#include <cstdlib>

#include <algorithm>

#define repu(i,x,y) for (int i=x; i<=y; ++i)

using namespace std;


typedef long long LL;

int n,m,q;

struct node

{

    int pri,w,s;

    LL l,r;

    node *lc,*rc;

    void update()

    {

        s=lc->s+rc->s+w;

    }

} pool[3000000],*tp=pool,*nul=tp;

struct data

{

    node *root;

    void rotate_l(node *&x)
    {
        node *y=x->rc;
        x->rc=y->lc,y->lc=x;
        x->update(),y->update(),x=y;
    }
    void rotate_r(node *&x)
    {
        node *y=x->lc;
        x->lc=y->rc,y->rc=x;
        x->update(),y->update(),x=y;
    }
    void insert(node *&x,LL l,LL r,bool flag);
    LL erase(node *&x,int k);
    void init();
    void insert(LL l,LL r);
    LL erase(int k);
} a[300100],b;

int getint()
{
    char ch;
    while (!isdigit(ch=getchar()));
    int x=ch-'0';
    for (; isdigit(ch=getchar()); x=x*10+ch-'0');
    return x;
```

```
}

void data::init()
{
    root=nul;
}

void data::insert(node *&x,LL l,LL r,bool flag)
{
    if (x==nul)
    {
        (x=++tp)->pri=rand();
        x->l=l,x->r=r,x->w=x->s=r-l+1;
        x->lc=x->rc=nul;
        return;
    }
    if (flag)
    {
        insert(x->rc,l,r,flag),x->update();
        if (x->rc->pri<x->pri)
            rotate_l(x);
    }
    else
    {
        insert(x->lc,l,r,flag),x->update();
        if (x->lc->pri<x->pri)
            rotate_r(x);
    }
}

void data::insert(LL l,LL r)
{
    insert(root,l,r,1);
}

LL data::erase(node *&x,int k)
{
    if (k>x->lc->s && k<=x->lc->s+x->w)
    {
        LL ret=x->l+k-x->lc->s-1;
        if (ret<x->r)
            insert(x->rc,ret+1,x->r,0);
        x->r=ret-1,x->w=x->r-x->l+1,x->update();
        return ret;
    }
    if (k<=x->lc->s)
    {
        LL ret=erase(x->lc,k);
        x->update();
        if (x->lc->pri<x->pri)
```

```
            rotate_r(x);
        return ret;
    }
    else
    {
        LL ret=erase(x->rc,k-x->lc->s-x->w);
        x->update();
        if (x->rc->pri<x->pri)
            rotate_l(x);
        return ret;
    }
}

LL data::erase(int k)
{
    erase(root,k);
}

int main()
{
    freopen("phalanx.in","r",stdin);
    freopen("phalanx.out","w",stdout);
    scanf("%d%d%d",&n,&m,&q);
    repu(i,1,n)
        a[i].init(),a[i].insert(LL(i-1)*m+1,LL(i)*m-1);
    b.init();
    repu(i,1,n)
        b.insert(LL(i)*m,LL(i)*m);
    while (q--)
    {
        int x=getint(),y=getint();
        if (y==m)
        {
            LL t=b.erase(x);
            b.insert(t,t);
            printf("%lld\n",t);
        }
        else
        {
            LL t1=a[x].erase(y),t2=b.erase(x);
            a[x].insert(t2,t2),b.insert(t1,t1);
            printf("%lld\n",t1);
        }
    }
    return 0;
}
```

蔡承泽

```
#include<cstdio>
const int N=300007;
int read(){
	int x=0,c=getchar();
	while(c<48)c=getchar();
	while(c>47)x=x*10+c-48,c=getchar();
	return x;
}
typedef long long i64;
void pr(i64 x){
	int ss[20],sp=0;
	do ss[++sp]=x%10;while(x/=10);
	while(sp)putchar(ss[sp--]+'0');
	putchar(10);
}
#define lc c[0]
#define rc c[1]
int n,m,q;
struct node{
	node*c[2];
	int D,sz,rnd;
	i64 id;
//	void pr(){
//		if(!this)return;
//		if(lc&&lc->rnd>rnd)D/=0;
//	    if(rc&&rc->rnd>rnd)D/=0;
//	    lc->pr();
//	    rc->pr();
//	}
	void up(){
		sz=D;
		if(lc)sz+=lc->sz;
		if(rc)sz+=rc->sz;
	}
}ns[N*10],*np=ns,*rt[N];
int seed=241255441;
int myrand(){
	return seed=(seed*184467711+12347)%1000000007;
}
node*newnode(i64 L,i64 R){
	if(L>R)return 0;
	node*w=np++;
	w->lc=w->rc=0;
	w->D=w->sz=R-L+1;
	w->rnd=myrand();
	w->id=L;
	return w;
}
#define $(a,b,c) (*a=b,a=&c,b=*a)
void mg(node*a,node*b,node**res){
        while(a&&b){
                if(a->rnd>b->rnd){
                        a->sz+=b->sz;
                        $(res,a,a->rc);
                }else{
                        b->sz+=a->sz;
                        $(res,b,b->lc);
                }
        }
        *res=a?a:b;
}
node*dd;
node*chk(node*w,int d){
        node*u=w->c[d];
        if(u&&u->rnd>w->rnd){
                w->c[d]=u->c[d^1];
                u->c[d^1]=w;
                w->up(),u->up();
                return u;
        }
        return w;
}
node*del(node*w,int k){
        int ls=w->lc?w->lc->sz:0;
        if(k<=ls){
```

```
		--w->sz;
		w->lc=del(w->lc,k);
		return chk(w,0);
	}
	k-=ls;
	if(k>w->D){
		--w->sz;
		w->rc=del(w->rc,k-w->D);
		return chk(w,1);
	}
	i64 p=w->id+k-1;
	dd=newnode(p,p);
	node*l=newnode(w->id,p-1);
	node*r=newnode(p+1,w->id+w->D-1);
	if(myrand()%10<5){
		if(l)l->rnd=w->rnd;
	}else if(r)r->rnd=w->rnd;
	mg(w->lc,l,&l);
	mg(r,w->rc,&r);
	mg(l,r,&w);
	return w;
}
void run(){
	n=read();m=read();q=read();
	for(int i=1;i<=n;++i){
			i64 p=m*i64(i-1);
			rt[i]=newnode(p+1,p+m-1);
			mg(rt[0],newnode(p+m,p+m),&rt[0]);
		}
		for(int i=1;i<=q;++i){
			int x=read(),y=read();
			node*p1,*p2;
			if(y==m){
				rt[0]=del(rt[0],x);
				pr(dd->id);
				mg(rt[0],dd,&rt[0]);
			}else{
				rt[0]=del(rt[0],x);p1=dd;
				rt[x]=del(rt[x],y);p2=dd;
				pr(p2->id);
				mg(rt[x],p1,&rt[x]);
				mg(rt[0],p2,&rt[0]);
			}
		}
	}
	int main(){
		freopen("phalanx.in","r",stdin);
		freopen("phalanx.out","w",stdout);
		run();
		fclose(stdin);
	fclose(stdout);
	return 0;
}
```

马耀华

```
#include <cstdio>
#include <cstring>
#include <algorithm>
#define MOD 1000000007

using namespace std;

typedef long long ll;

ll now=123;

int rd() {
  now=now*197043%MOD;
  return now;
}

int ch[3000000][2],a[3000000],tot;
ll num[3000000];
```

```
int len[3000000],size[3000000];

int rotate(int x,int y) {
  int d=(ch[y][1]==x);
  ch[y][d]=ch[x][d^1];ch[x][d^1]=y;
  size[y]=size[ch[y][0]]+size[ch[y][1]]+len[y];
  size[x]=size[ch[x][0]]+size[ch[x][1]]+len[x];
  return x;
}

int insert(int x,int k,ll p,int q) {
  if (!x) {
      x=++tot;
      a[x]=rd();
      num[x]=p;len[x]=size[x]=q;
      return x;
  }
  int d=(size[ch[x][0]]>=k)?0:1;
  if (!d) ch[x][d]=insert(ch[x][d],k,p,q);
  else ch[x][d]=insert(ch[x][d],k-size[ch[x][0]]-len[x],p,q);
  if (a[ch[x][d]]<a[x]) x=rotate(ch[x][d],x);
  size[x]=size[ch[x][0]]+size[ch[x][1]]+len[x];
  return x;
}
```

```
int ans1,ans2;


int erase(int x,int k,bool v) {
  if ((size[ch[x][0]]<k&&size[ch[x][0]]+len[x]>=k)||v) {
       if (!v) {
       ans1=x;ans2=k-size[ch[x][0]];
    }
       if (!ch[x][0]) return ch[x][1];
       if (!ch[x][1]) return ch[x][0];
       int d=(a[ch[x][0]]>a[ch[x][1]]);
       x=rotate(ch[x][d],x);
       ch[x][d^1]=erase(ch[x][d^1],k,1);
       size[x]=size[ch[x][0]]+size[ch[x][1]]+len[x];
       return x;
  }
  else {
       if (size[ch[x][0]]>=k) ch[x][0]=erase(ch[x][0],k,0);
       else ch[x][1]=erase(ch[x][1],k-size[ch[x][0]]-len[x],0);
       size[x]=size[ch[x][0]]+size[ch[x][1]]+len[x];
       return x;
  }
}


int root[300005],rt,n,m;
```

```
ll update1(int x) {

  ans1=ans2=0;

  rt=erase(rt,x,0);

  rt=insert(rt,n-1,num[ans1],len[ans1]);

  return num[ans1];

}


ll update2(int x,int y) {

  ans1=ans2=0;

  root[x]=erase(root[x],y,0);

  int p=y-ans2;

  ll q=num[ans1]+ans2-1;

  if (ans2>1) root[x]=insert(root[x],p,num[ans1],ans2-1);

  if (ans2<len[ans1]) root[x]=insert(root[x],y-1,num[ans1]+ans2,len[ans1]-
ans2);

  ans1=ans2=0;

  rt=erase(rt,x,0);

  rt=insert(rt,n-1,q,1);

  root[x]=insert(root[x],m-2,num[ans1],len[ans1]);

  return q;

}


int main() {

  freopen("phalanx.in","r",stdin);

  freopen("phalanx.out","w",stdout);

  int k;
```

```
    scanf("%d%d%d",&n,&m,&k);
    for(int i=1;i<=n;i++) root[i]=insert(root[i],0,(ll)(i-1)*m+1,m-1);
    for(int i=1;i<=n;i++) rt=insert(rt,i-1,(ll)i*m,1);
    for(int i=1;i<=k;i++) {
        int x,y;
        scanf("%d%d",&x,&y);
        if (y==m) printf("%lld\n",update1(x));
        else printf("%lld\n",update2(x,y));
    }
    return 0;
}
```