Michal Korbela
septima
Gym. J.J. Bánovce nad Bebravou
úloha 4

Túto úlohu môžeme riešiť simulovaním situácie.
Ako si však zapamätám danú situáciu ??
Pre každé políčko na pláne si budem pamätať ktorá korytnačka je na vrchu a pre každú korytnačku si budem pamätať ktorá korytnačka je pod a ktorá je nad ňou. Ak žiadna korytnačka tam nebude, tak tam zapíšem 0, pretože takáto korytnačka neexituje , takže 0 môžem použiť ako konštantu ničoho.

Potom si načítam korytnačky a nastavím im ich korytnačku pod a nad .

Teraz máme nastavený počiatočný stav. Môžeme ísť posúvať.

Keď posunieme nejakú korytnačku k, tak jedniné čo sa zmení, je korytnačka ktorá je nad korytnačkov ktorá je pod korytnačkov k,
na vrchu políčka potom bude korytnačka ktorá je pod korytnačkov k
Na novom políčku potom na vrchu políčka bude korytnačka ktorá bola na vrchu predchádzajúceho políčka, korytnačka na vrchu nového políčka (ešte pred posunutím) bude mať nad sebou korytnačku k a korytnačka k zas ju.

Samozrejme ak nieje pod korytnačkou žiadna korytnačka tak sa nastaví pod ňu 0.

K zložitostiam

čas, musíme načítať všetky korytnačky čo nám trvá O(k)
potom pre každý ťah musíme spraviť konštantný počet krokov čo je O(t)
a potom musíme ešte vypísať p políčok s k korytnačkami čo nám trvá O(k+p) času
a potom výsledný čas je O(k+t+p)

Pamäť – keďže pre každú korytnačku a políčko si potrebujem pamätať konštantne veľa, tak potom pamäť bude O(k+p)

```
#include<iostream>
#include<cstdio>

using namespace std;


struct str{                    //štruktúra pre korytnačku a políčko(aj keď v ňom nevyužijeme pod)

int nad;
int pod;

};


str kor[300000];               // pamätáme si všetky korytnačky
```

```
str plan[300000];               //pamätáme si políčka


int main(){
int k,t,p,prev;


cin>>k>>p>>t;                   // načítame vstup

for(int i=1; i<=p; i++)         //vynulujeme plán
{
plan[i].nad=0;
plan[i].pod=0;
}

for (int i=1; i<=k; i++){       // načítame korytnačky
int l;
scanf("%d",&l);
if(i==1){
kor[l].pod=0;                   //nastvíme im čo majú pod a čo nad
plan[1].pod=l;
}
else
kor[l].pod=prev;

if(i>1)
kor[prev].nad=l;

if(i==k)                        // vrchná korytnačka nemá nad sebou nič
{
kor[l].nad=0;
plan[1].nad=l;                  //na vrchu plánu je táto posledná korytnačka

}
prev=l;                         // pamätáme si poslednú porytnačku
}

/////nacitanie;


for(int i=1; i<=t; i++){
int tur,policko,tah,nad,pod;    // vykonávame ťahy



//cin>>tur>>policko>>tah;       // načítame ťah
scanf("%d",&tur);
scanf("%d",&policko);
scanf("%d",&tah);
```

```
nad=kor[tur].nad;                    // zalohujeme si stav
pod=kor[tur].pod;
int plan_nad=plan[policko].nad;

                                     //odpojime korytnacku
if(kor[tur].pod>0){
kor[kor[tur].pod].nad=0;             // nastavíme 0 pod a nad korytnačkami
kor[tur].pod=0;
}
if(tur==plan[policko].pod)           // ak pod kotytnačkou nič nieje
{
plan[policko].pod=0;                 // aj na vrchu políčka nič nebude
plan[policko].nad=0;
}
else{

plan[policko].nad=pod;               // inak tam bude korytnačka pod zobratou

}




//pripojime korytnacku
if(plan[policko+tah].nad==0){              //ak je policko prazdne
plan[policko+tah].pod=tur;           //na spodok dame aktualnu
plan[policko+tah].nad=plan_nad;            //na vrch dame vrchnu zobratu
kor[tur].pod=0;                            //pod prvou nic nieje
}
else
{
kor[plan[policko+tah].nad].nad=tur;        //nad poslednu na novom mieste ju dame
kor[tur].pod=plan[policko+tah].nad;        //pod aktualnu dame poslednu

plan[policko+tah].nad=plan_nad;            // na vrch dame vrchnu
}


}




for(int i=1; i<=p; i++)                    // spočítame korytančky
{
int pocet=0;
int next=plan[i].pod;
while(next!=0){
```

```
pocet++;


next=kor[next].nad;




}

cout<<pocet;
//////////////////////////////////////////////////////


next=plan[i].pod;
while(next!=0){                                  //vypíšeme ich



cout<<" "<<next;


next=kor[next].nad;




}
cout<<"\n";


}

}
```