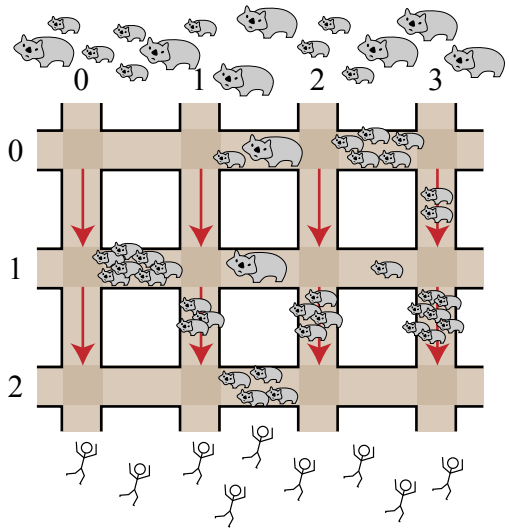
International Olympiad in Informatics 2013
6-13 July 2013
Brisbane, Australia

Sysle

Slovak — 1.0

Mesto Brisbane bolo napadnuté veľkou skupinou zmutovaných Sysľov. Pôvodne mala mesto evakuovať Stará Raketa, ale akosi sa pokazila a úloha evakuovať mesto ostala na vás.

Cesty v Brisbane tvoria veľkú mriežku. Nachádza sa tam `R` horizontálnych ciest, ktoré spájajú východ a západ a majú čísla `0, …, (R - 1)` v poradí od severu na juh, a `C` vertikálnych ciest, ktoré spájajú sever a juh a majú čísla `0, …, (C - 1)` v poradí od západu na východ, viď obrázok nižšie.



Sysle napadli mesto zo severu a ľudia utekajú na juh. Ľudia môžu utekať po horizontálnych cestách ľubovoľným smerom, ale po vertikálnych cestách sú ochotní utekať iba do bezpečia *smerom na juh*.

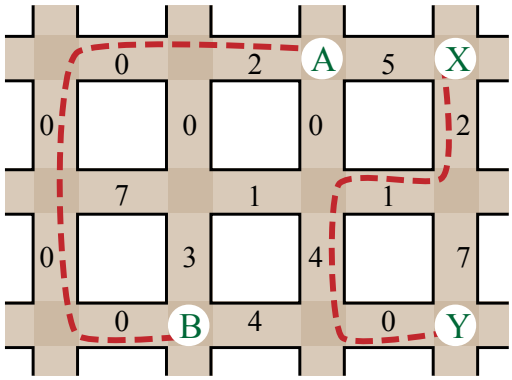
Križovatku medzi horizontálnou cestou `P` a vertikálnou cestou `Q` budeme označovať ako `(P, Q)`. Každý kus cesty medzi dvoma križovatkami obsahuje nejaký počet Sysľov a tento počet sa môže v priebehu času meniť. Vašou úlohou je navigovať každého človeka z niektorej konkrétnej križovatky na severe (na horizontálnej ceste `0`) ku niektorej konkrétnej križovatke na juhu (na horizontálnej ceste `R-1`) takou cestou, ktorá obsahuje najmenší možný počet Sysľov.

Na začiatku dostanete rozmery mriežky a počet Sysľov na každom kuse cesty. Nasledovne dostanete sériu `E` udalostí, pričom každá z nich je jedného z nasledovných typov:

- *zmena*, ktorá zmení počet Sysľov na danom kuse cesty
- *útek*, kde sa osoba zjaví na danej križovatke na horizontálnej ceste `0` a vy musíte nájsť cestu na danú križovatku na horizontálnej ceste `R - 1`, na ktorej sa nachádza najmenší možný počet Sysľov.

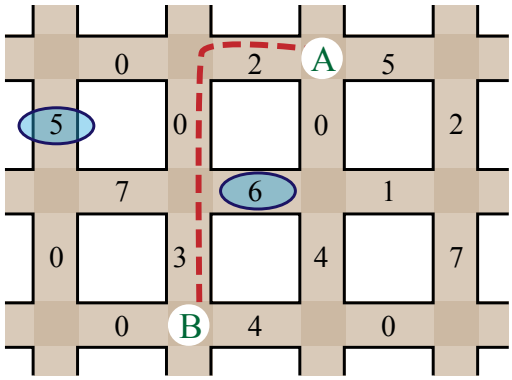
Vašou úlohou je spracovať tieto udalosti implementovaním funkcií `init()`, `changeH()`, `changeV()` a `escape()`, tak ako je popísané nižšie.

Príklady



Obrázok vyššie ukazuje počiatočnú mriežku, kde máme `R = 3` horizontálnych ciest a `C = 4` vertikálnych ciest. Pri každom kuse cesty je vyznačený počet Sysľov. Uvažujme nasledujúcu postupnosť udalostí:

- Človek sa zjaví na križovatke `A = (0, 2)` a chce utiecť ku križovatke `B = (2, 1)`. Najmenší možný počet Sysľov, ktorých musí stretnúť je `2`, ako je vyznačené prerušovanou čiarou.
- Ďalší človek sa zjaví na križovatke `X = (0, 3)` a chce utiecť ku križovatke `Y = (2, 3)`. Najmenší možný počet Sysľov, ktorých musí stretnúť je `7`, ako je opäť vyznačené prerušovanou čiarou.
- Nastanú dve zmeny: počet Sysľov na najvyššom kuse vertikálnej cesty `0` sa zmení na `5` a počet Sysľov na strednom kuse horizontálnej cesty `1` sa zmení na `6`. Viď zakrúžkované čísla na obrázku nižšie.



- Tretí človek sa objaví na križovatke `A = (0, 2)` a chce utiecť ku križovatke `B = (2, 1)`. Teraz je najmenší počet Sysľov, cez ktorých musí prejsť `5`, ako je vyznačené novou prerušovanou čiarou.

Implementácia

Máte odovzdať súbor, v ktorom implementujete procedúry: `init()`, `changeH()` a `changeV()` a funkciu `escape()`, ktoré sú popísané nižšie.

Vaša procedúra: `init()`

C/C++
```
void init(int R, int C, int H[5000][200], int V[5000][200]);
```

Pascal
```
type wombatsArrayType = array[0..4999, 0..199] of LongInt;

procedure init(R, C : LongInt; var H, V : wombatsArrayType);
```

Popis

Táto procedúra vám zadá počiatočné rozloženie mriežky a umožní vám inicializovať globálne premenné a dátové štruktúry. Bude zavolaná len raz, pred všetkými volaniami funkcií `changeH()`, `changeV()` alebo `escape()`.

Parametre

- `R`: Počet horizontálnych ciest.
- `C`: Počet vertikálnych ciest.
- `H`: Dvojrozmené pole veľkosti `R × (C - 1)`, kde `H[P][Q]` udáva počet Sysľov na kuse horizontálnej cesty medzi križovatkami `(P, Q)` a `(P, Q + 1)`.
- `V`: Dvojrozmerné pole veľkosti `(R - 1) × C`, kde `V[P][Q]` udáva počet Sysľov na kuse vertikálnej cesty medzi križovatkami `(P, Q)` a `(P + 1, Q)`.

Vaša procedúra: `changeH()`

C/C++
```
void changeH(int P, int Q, int W);
```

Pascal
```
procedure changeH(P, Q, W: LongInt);
```

Popis

Táto procedúra bude zavolaná, keď sa zmení počet Sysľov na kuse horizontálnej cesty medzi križovatkami (P, Q) a $(P, Q + 1)$.

Parametre

- `P`: Označuje, ktorej horizontálnej cesty sa zmena týka ($0 \le P \le R - 1$).
- `Q`: Označuje medzi ktorými dvoma vertikálnymi cestami zmena nastala ($0 \le Q \le C - 2$).
- `W`: Nový počet Sysľov na danom kuse cesty ($0 \le W \le 1{,}000$).

Vaša procedúra: `changeV()`

C/C++ `void changeV(int P, int Q, int W);`

Pascal `procedure changeV(P, Q, W: LongInt);`

Popis

Táto procedúra bude zavolaná, keď sa zmení počet Sysľov na kuse vertikálnej cesty medzi križovatkami (P, Q) a $(P+1, Q)$.

Parametre

- `P`: Označuje medzi ktorými dvoma horizontálnymi cestami zmena nastala ($0 \le P \le R - 2$).
- `Q`: Označuje, ktorej vertikálnej cesty sa zmena týka ($0 \le Q \le C - 1$).
- `W`: Nový počet Sysľov na danom kuse cesty ($0 \le W \le 1{,}000$).

Vaša funkcia: `escape()`

C/C++ `int escape(int V1, int V2);`

Pascal `function escape(V1, V2 : LongInt) : LongInt;`

Popis

Táto funkcia má spočítať najmenší počet Sysľov, okolo ktorých musí človek prejsť, ak chce prejsť z križovatky $(0, V1)$ do križovatky $(R-1, V2)$.

Parametre

- `V1`: Označuje, kde človek začína na riadku 0 ($0 \le V1 \le C-1$).
- `V2`: Označuje, kde človek skončí na riadku $R-1$ ($0 \le V2 \le C-1$).
- *Vracia*: Najmenší možný počet Sysľov, okolo ktorých musí človek prejsť.

Ukážkový beh

Nasledujúca séria volaní popisuje príklad uvedený vyššie:

Function Call	Returns
`init(3, 4, [[0,2,5], [7,1,1], [0,4,0]], [[0,0,0,2], [0,3,4,7]])`	
`escape(2,1)`	`2`
`escape(3,3)`	`7`
`changeV(0,0,5)`	
`changeH(1,1,6)`	
`escape(2,1)`	`5`

Obmedzenia

- Časový limit: 20 sekúnd
- Pamäťový limit: 256 MiB
- $2 \le R \le 5{,}000$
- $1 \le C \le 200$
- Najviac 500 zmien (volaní buď `changeH()` alebo `changeV()`)
- Najviac 200,000 volaní `escape()`
- Najviac 1,000 Sysľov na jednom kuse cesty v ľubovoľnom čase.

Podúlohy

Podúloha	Body	Ďalšie obmedzenia vstupu
1	9	$C = 1$
2	12	$R,C \le 20$, a žiadne volania `changeH()` alebo `changeV()`
3	16	$R,C \le 100$, a najviac 100 volaní `escape()`
4	18	$C = 2$
5	21	$C \le 100$
6	24	*(Žiadne)*

Experimentovanie

Ukážkový grader na vašom počítači bude čítať zo súboru `wombats.in`, ktorý musí byť v nasledovnom formáte:

- riadok 1: `R C`
- riadok 2: `H[0][0] … H[0][C-2]`
- …
- riadok $(R + 1)$: `H[R-1][0] … H[R-1][C-2]`
- riadok $(R + 2)$: `V[0][0] … V[0][C-1]`
- …
- riadok $(2R)$: `V[R-2][0] … V[R-2][C-1]`
- ďalší riadok: `E`
- ďalších `E` riadkov: popisy udalostí v poradí, v akom sa vyskytnú (jedna v každom riadku)

Ak $C = 1$, prázdne riadky obsahujúce počet Sysľov na horizontálnych cestách (riadky `2` až `R + 1`) nie sú potrebné.

Riadok pre každú udalosť musí byť v jednom z nasledujúcich formátov:

- pre `changeH(P, Q, W)`: `1 P Q W`
- pre `changeV(P, Q, W)`: `2 P Q W`
- pre `escape(V1, V2)`: `3 V1 V2`

Napríklad, ukážka vyššie by mala nasledovný formát:

```
3 4
0 2 5
7 1 1
0 4 0
0 0 0 2
0 3 4 7
5
3 2 1
3 3 3
2 0 0 5
1 1 1 6
3 2 1
```

Poznámky k použitému programovaciemu jazyku

C/C++ Váš súbor musí obsahovať `#include "wombats.h"`.

Pascal Musíte definovať `unit Wombats`. Polia sú číslované od `0` (nie od `1`).

Viď. predlohy riešení na vašich počítačoch.