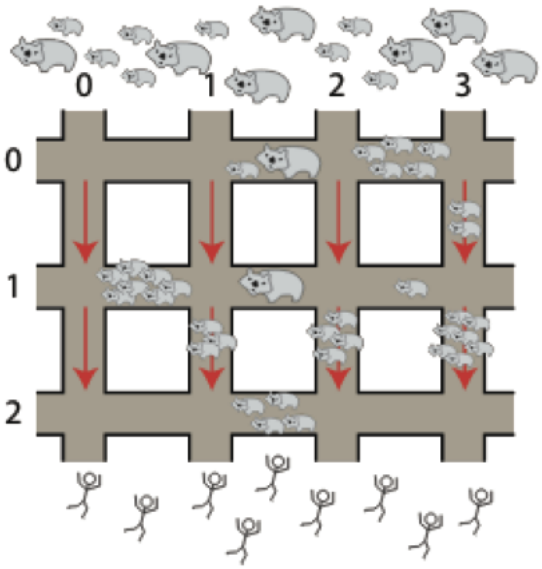
Brisbane so zavzeli veliki mutirani vombati (avstralski vrečarji). Tvoja naloga je popeljati ljudi na varno.

Ceste v Brisbaneu so razporejene v pravokotno mrežo. V smeri od vzhoda proti zahodu poteka `R` vodoravnih cest, označenih s številkami `0, ..., (R - 1)` po vrsti od severa proti jugu. Pravokotno nanje poteka `C` navpičnih cest, označenih s številkami `0, ..., (C - 1)` po vrsti od vzhoda proti zahodu. Oglej si sledečo sliko:



Ker so vombati napadli s severa, ljudje bežijo proti jugu. Ljudje lahko po vodoravnih cestah bežijo v obeh smereh, po navpičnih pa *samo proti jugu* (varnosti naproti).

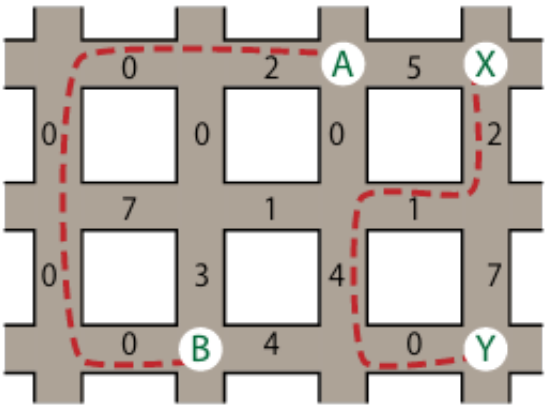
Križišče vodoravne ceste `P` in navpične ceste `Q` je označeno kot `(P, Q)`. Vsak odsek ceste med sosednjima križiščema vsebuje določeno število vombatov, ki pa se s časom lahko spreminja. Tvoja naloga je popeljati vsako osebo s podanega križišča na skrajnem severu (na vodoravni cesti 0) do podanega križišča na skrajnem jugu (na vodoravni cesti `R-1`), in sicer po poti, ki vsebuje najmanjše možno število vombatov.

Na začetku bo podana velikost mreže cest in število vombatov na vsakem posameznem cestnem odseku. Nato bo podano zaporedje `E` dogodkov sledečih dveh tipov:

- *sprememba*: Označuje spremembo števila vombatov na nekem cestnem odseku.
- *pobeg*: Označuje pobeg neke osebe s podanega križišča na vodoravni cesti 0 do podanega križišča na vodoravni cesti `R-1` po poti, ki vsebuje najmanjše število vombatov.

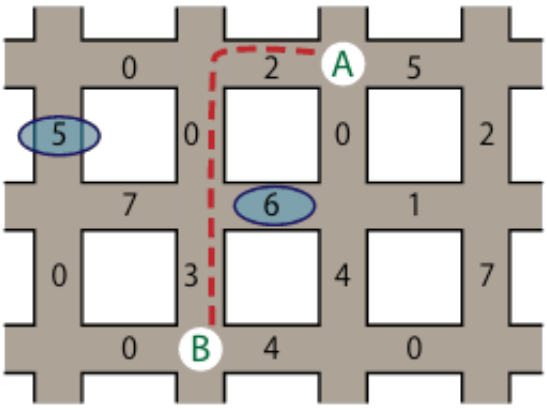
Za obravnavo opisanih tipov dogodkov napiši procedure `init()`, `changeH()` in `changeV()` ter funkcijo `escape()` v skladu z navodili, opisanimi v nadaljevanju.

Primeri



Gornja slika prikazuje začetni zemljevid z `R = 3` vodoravnimi in `C = 4` navpičnimi cestami. Za vsak cestni odsek je navedeno število vombatov. Oglejmo si sledeče zaporedje dogodkov:

- Oseba prispe na križišče `A = (0, 2)` in želi pobegniti na križišče `B = (2, 1)`. Kot nakazuje leva črtkana črta, je najmanjše možno število vombatov, ki jih sreča na poti, enako 2.
- Neka druga oseba prispe na križišče `X = (0, 3)` in želi pobegniti na križišče `Y = (2, 3)`. Najmanjše možno število vombatov, ki jih sreča na poti, je tokrat enako 7 (desna črtkana črta).
- Zgodita se dve spremembi: število vombatov na zgornjem odseku navpične ceste 0 se spremeni na 5, število vombatov na srednjem odseku vodoravne ceste 1 pa se spremeni na 6 (glej obkroženi številki na spodnji sliki).



- Tretja oseba prispe na križišče `A = (0, 2)` in želi pobegniti do križišča `B = (2, 1)`. Kot nakazuje črtkana črta, je najmanjše možno število vombatov na njeni poti enako `5`.

Implementacija

Oddaj datoteko, v kateri so po sledečih navodilih implementirane procedure `init()`, `changeH()` in `changeV()` ter funkcija `escape()`:

Tvoja procedura: `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);
```

Opis

Ta procedura sprejme začetni zemljevid. V njej lahko inicializiraš poljubne globalne spremenljivke in podatkovne strukture. Klicala se bo le enkrat, in sicer pred klici procedur `changeH()` in `changeV()` ter funkcije `escape()`.

Parametri

- `R`: Število vodoravnih cest.
- `C`: Število navpičnih cest.
- `H`: Dvodimenzionalno polje velikosti $R \times (C - 1)$, v katerem element `H[P][Q]` vsebuje število vombatov na vodoravnem odseku med križiščema (P, Q) in $(P, Q + 1)$.
- `V`: Dvodimenzionalno polje velikosti $(R - 1) \times C$, v katerem element `V[P][Q]` vsebuje število vombatov na navpičnem odseku med križiščema (P, Q) in $(P + 1, Q)$.

Tvoja procedura: `changeH()`

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

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

Opis

Ta procedura se kliče ob spremembi števila vombatov na vodoravnem odseku med križiščema (P, Q) in $(P, Q + 1)$.

Parametri

- `P` : Določa, na kateri vodoravni cesti se nahaja obravnavani odsek ($0 \le P \le R - 1$).
- `Q` : Določa, med katerima navpičnima cestama se nahaja obravnavani odsek ($0 \le Q \le C - 2$).
- `W` : Novo število vombatov na obravnavanem odseku ($0 \le W \le 1000$).

Tvoja procedura: `changeV()`

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

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

Opis

Ta procedura se kliče ob spremembi števila vombatov na navpičnem odseku med križiščema (P, Q) in $(P+1, Q)$.

Parametri

- `P` : Določa, med katerima vodoravnima cestama se nahaja obravnavani odsek ($0 \le P \le R - 2$).
- `Q` : Določa, na kateri navpični cesti se nahaja obravnavani odsek ($0 \le Q \le C - 1$).
- `W` : Novo število vombatov na obravnavanem odseku ($0 \le W \le 1000$).

Tvoja funkcija: `escape()`

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

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

Opis

Ta funkcija naj izračuna najmanjše možno število vombatov, ki jih oseba sreča na poti od križišča $(0, V1)$ do $(R-1, V2)$.

Parametri

- `V1` : Določa, kje na vodoravni cesti 0 se oseba nahaja na začetku ($0 \le V1 \le C-1$).
- `V2` : Določa, kje na vodoravni cesti $R-1$ ($0 \le V2 \le C-1$) oseba zaključi svojo pot.
- *Vrača*: Najmanjše število vombatov, ki jih oseba sreča na svoji poti.

Vzorčno zaporedje klicev

Gornji primer je opisan s sledečim zaporedjem klicev:

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`

Omejitve

- Časovna omejitev: 20 sekund
- Prostorska omejitev: 256 MiB
- $2 \le R \le 5000$
- $1 \le C \le 200$
- Največ 500 sprememb (klicev procedur `changeH()` in `changeV()`)
- Največ 200 000 klicev funkcije `escape()`
- Največ 1000 vombatov na poljubnem odseku v poljubnem trenutku.

Podnaloge

Podnaloga	Točke	Dodatne vhodne omejitve
1	9	$C = 1$
2	12	$R,C \le 20$, klicev procedur `changeH()` in `changeV()` pa ne bo
3	16	$R,C \le 100$, klicev funkcije `escape()` pa bo največ 100
4	18	$C = 2$
5	21	$C \le 100$
6	24	*(Brez)*

Preizkušanje

Vzorčni ocenjevalnik na tvojem računalniku bere vhod iz datoteke `wombats.in`, ki mora biti v sledečem formatu:

- vrstica 1: `R C`
- vrstica 2: `H[0][0] … H[0][C-2]`
- …
- vrstica `(R + 1)`: `H[R-1][0] … H[R-1][C-2]`
- vrstica `(R + 2)`: `V[0][0] … V[0][C-1]`
- …
- vrstica `(2R)`: `V[R-2][0] … V[R-2][C-1]`
- naslednja vrstica: `E`
- naslednjih `E` vrstic: po en dogodek na vrstico (dogodki so podani v vrstnem redu njihovega nastopanja).

Če je `C = 1`, potem so vrstice od `2` do `R+1` prazne, zato niso obvezne.

Vrstice, ki opisujejo posamezne dogodke, so v eni od sledečih oblik:

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

Gornji primer, denimo, je podan v sledečem formatu:

```
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
```

Jezikovne opombe

C/C++	Potrebuješ `#include "wombats.h"`.
Pascal	Definiraj `unit Wombats`. Oštevilčenje vseh polj se prične z `0` in ne z `1`.

Za primer glej predlogo rešitve na svojem računalniku.