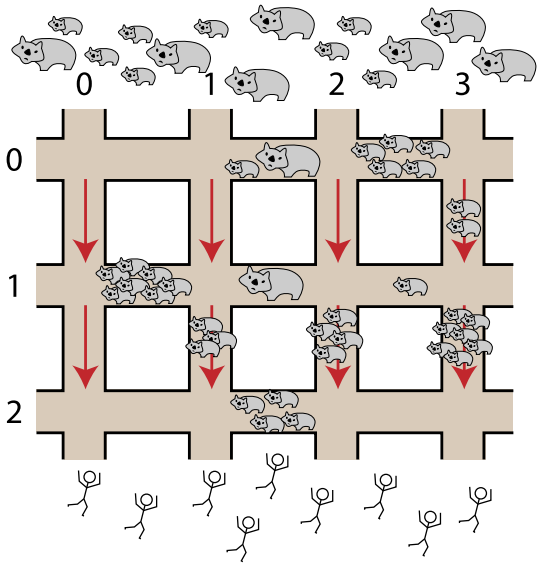
International Olympiad in Informatics 2013
6-13 July 2013
Brisbane, Australia

wombats

Spanish-VE — 1.0

La ciudad de Brisbane ha sido invadida por una gran cantidad de wombats mutantes, y usted debe llevar a la gente a salvo.

Las calles en Brisbane están dispuestas en una gran red. Existen `R` calles horizontales que van de este a oeste, numeradas `0, ..., (R - 1)` en orden de norte a sur, y `C` calles verticales que van de norte a sur, numeradas `0, ..., (C - 1)` en orden de oeste a este, como se muestra en la figura.



Los wombats han invadido desde el norte, y las personas están escapando hacia el sur. Las personas pueden correr a través de las calles horizontales en cualquier dirección, pero en las calles verticales ellos *sólo pueden correr hacia el sur*, buscando ponerse a salvo.

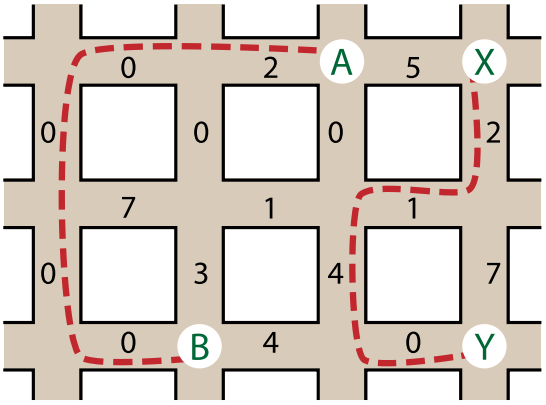
Las intersecciones entre una calle horizontal `P` con una calle vertical `Q` se denota `(P, Q)`. Cada segmento de calle entre dos intersecciones contiene un número de wombats, y esos números pueden cambiar en el tiempo. Su tarea es guiar a cada persona desde una intersección dada en el norte (en la calle horizontal `0`) a alguna intersección en el sur (en la calle horizontal `R - 1`), llevandolos por una ruta donde pase por la menor cantidad de wombats.

Para comenzar, a usted se le proveerá del tamaño de la red y el número de wombats en cada calle. Luego de esto, usted recibirá una serie de `E` eventos, cada uno de los cuales es:

- un *change*, el cual, cambia el número de wombats en un segmento; ó
- un *escape*, donde una persona llega a una intersección dada en la calle horizontal `0`, y luego usted debe encontrar una ruta a una intersección dada en la calle horizontal `R - 1` pasando por la menor cantidad posible de wombats.

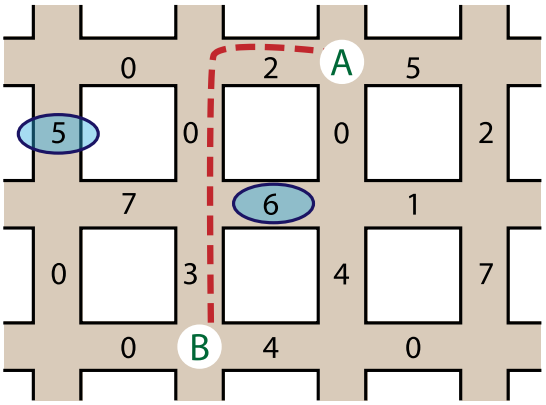
Usted debe manejar dichos eventos implementando las funciones `init()`, `changeH()`, `changeV()` and `escape()`, como se describen más adelante.

Ejemplos



La imagen anterior muestra una red inicial con `R = 3` calles horizontales y `C = 4` calles verticales, con el número de wombats marcado en cada segmento. Considere la siguiente serie de eventos:

- Una persona llega a la intersección `A = (0, 2)` y desea escapar a la intersección `B = (2, 1)`. El menor número de wombats por el que puede pasar es de `2`, como se indica por la línea punteada.
- Otra persona llega a la intersección `X = (0, 3)` y desea escapar a la intersección `Y = (2, 3)`. La menor cantidad de wombats por la que puede pasar es de `7`, de nuevo indicados por una línea punteada.
- Ocurren dos eventos de cambio: el número de wombats en el segmento superior de la calle vertical `0` cambia a `5`, y el número de wombats en el segmento central de la calle horizontal `1` cambia a `6`. Vea los números encerrados en círculos en la siguiente imagen.



- Una tercera persona llega a la intersección `A = (0, 2)` y desea escapar a la intersección `B = (2, 1)`. Ahora el menor número de wombats que puede pasar es `5`, como se indica por la línea punteada.

Implementación

Usted debe enviar un archivo que implemente los procedimientos `init()`, `changeH()` y `changeV()` y la función `escape()`, como sigue:

Su procedimiento: `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);
```

Descripción

Este procedimiento le provee la información inicial de las calles, y le permite inicializar cualquier variable global o estructura de datos. Este procedimiento será invocado una sola vez, antes de cualquier llamada a `changeH()`, `changeV()` o `escape()`.

Parámetros

- `R`: El número de calles horizontales.
- `C`: El número de calles verticales.
- `H`: Un arreglo bidimensional de tamaño `R × (C - 1)`, donde `H[P][Q]` contiene la cantidad de wombats en el segmento de calle horizontal entre las intersecciones `(P, Q)` y `(P, Q + 1)`.
- `V`: Un arreglo bidimensional de tamaño `(R - 1) × C`, donde `V[P][Q]` contiene el número de de wombats en el segmento de calle vertical entre las intersecciones `(P, Q)` y `(P + 1, Q)`.

Su procedimiento: `changeH()`

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

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

Descripción

Este procedimiento será invocado cuando el número de wombats cambie en el segmento de calle horizontal, entre las intersecciones `(P, Q)` y `(P, Q + 1)`.

Parámetros

- `P`: Indica cual calle horizontal es afectada (`0 ≤ P ≤ R - 1`).
- `Q`: Indica entre cuales dos calles verticales está el segmento (`0 ≤ Q ≤ C - 2`).
- `W`: El nuevo número de wombats en este segmento de calle (`0 ≤ W ≤ 1,000`).

Su procedimiento: `changeV()`

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

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

Descripción

Este procedimiento será invocado cuando el número de wombats cambie en el segmento de calle vertical, entre las intersecciones `(P, Q)` y `(P, Q + 1)`.

Parámetros

- `P`: Indica entre cuales calles horizontales se encuentra el segmento (`0 ≤ P ≤ R - 2`).
- `Q`: Indica cual calle vertical es afectada (`0 ≤ Q ≤ C - 1`).
- `W`: El nuevo número de wombats en este segmento de calle (`0 ≤ W ≤ 1,000`).

Tu Función: `escape()`

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

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

Descripción

Dicha función debe calcular la menor cantidad posible de wombats por los cuales debe pasar una persona viajando desde la intersección `(0, V1)` hasta `(R-1, V2)`.

Parámetros

- `V1`: Indica donde comienza la persona en la fila (`0 ≤ V1 ≤ C-1`).
- `V2`: Indica donde termina la persona en la fila `R-1` (`0 ≤ V2 ≤ C-1`).

- *Retorna*: El menor número de wombats que la persona debe pasar.

Sesión de Ejemplo

La siguiente sesión describe el ejemplo anterior:

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`

Restricciones

- Tiempo límite: 20 segundos
- Límite de memoria: 256 MiB (Megabytes)
- $2 \le R \le 5{,}000$
- $1 \le C \le 200$
- A lo sumo 500 cambios (llamadas a `changeH()` o `changeV()`)
- A lo sumo 200,000 llamadas a `escape()`
- A lo sumo 1,000 wombats en cualquier segmento, en cualquier momento.

Subtareas

Subtareas	Puntos	Condiciones de Entrada adicionales
1	9	$C = 1$
2	12	$R,C \le 20$, y no se realizarán llamadas a `changeH()` o `changeV()`
3	16	$R,C \le 100$, y se realizaran a lo sumo 100 llamadas a `escape()`
4	18	$C = 2$
5	21	$C \le 100$
6	24	*(None)*

Experimentación

El evaluador en su computadora leerá la entrada desde el archivo `wombats.in`, el cual, debe estar en el formato siguiente:

- línea 1: `R C`
- línea 2: `H[0][0] … H[0][C-2]`
- …
- línea `(R + 1)`: `H[R-1][0] … H[R-1][C-2]`
- línea `(R + 2)`: `V[0][0] … V[0][C-1]`
- …
- línea `(2R)`: `V[R-2][0] … V[R-2][C-1]`
- siguiente línea: `E`
- luego `E` líneas: un evento por línea, en el orden en el cual ocurren.

Si `C = 1`, las líneas vacías que contienen el número de wombats en las calles horizontales (líneas `2` hasta `R + 1`) no son necesarias.

La línea por cada evento debe estar en alguno de los siguientes formatos:

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

Por ejemplo, el caso mencionado en la descripción debe ser presentado en el siguiente formato:

```
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
```

Notas del lenguaje

C/C++	Usted debe `#include "wombats.h"`.
Pascal	Usted debe definir la `unit Wombats`. Todos los arreglos estan numerados a partir de `0` (no `1`).

Vea las plantillas de las soluciones en su computadora como ejemplo.