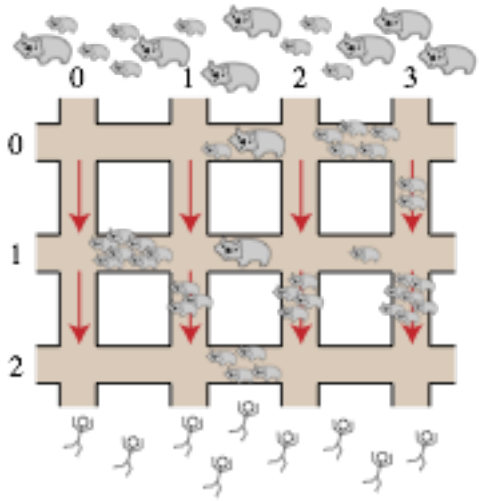
International Olympiad in Informatics
2013
6-13 July 2013
Brisbane, Australia

wombats

Español - ARG — 1.2

La ciudad de Brisbane ha sido capturada por grandes wombats mutados, y debes conducir la gente a lugar seguro.

Las calles en Brisbane forman una gran cuadrícula. En donde hay `R` calles horizontales que corren Este-a-Oeste, numeradas `0, ..., (R - 1)` ordenadas de Norte a Sur, y `C` calles verticales que corren Norte-a-Sur, numeradas `0, ..., (C - 1)` ordenadas de Oeste a Este, tal como se muestra en la figura siguiente:



Los wombats han invadido desde el Norte, y la gente está escapando hacia el Sur. La gente puede correr a lo largo de las calles horizontales en ambas direcciones, pero en las calles verticales *ellos correrán sólo hacia el Sur*, hacia un lugar seguro.

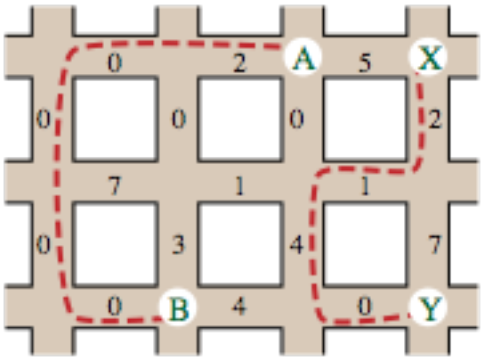
La intersección de la calle horizontal `P` con la calle vertical `Q` es denotada `(P, Q)`. Cada segmento de calle entre dos intersecciones contiene algún número de wombats, y estos números pueden variar a lo largo del tiempo. Tu tarea es guiar cada persona desde alguna intersección dada en el Norte (sobre la calle horizontal `0`) hacia alguna intersección dada en el Sur (sobre la calle horizontal `R - 1`), llevándolos sobre una ruta que pase el menor número de wombats como sea posible.

Para empezar, te será dado el tamaño de la cuadrícula y el número de wombats en cada segmento de calle . Después te será dada una serie de `E` eventos, cada uno de los cuales puede ser:

- un *change*, que altera el número de wombats en algún segmento de calle; o
- un *escape*, donde alguna persona llega a una intersección dada en la calle horizontal `0`, y debes encontrar una ruta hacia una intersección dada situada sobre la calle horizontal `R - 1` que pase el menor número posible de wombats.

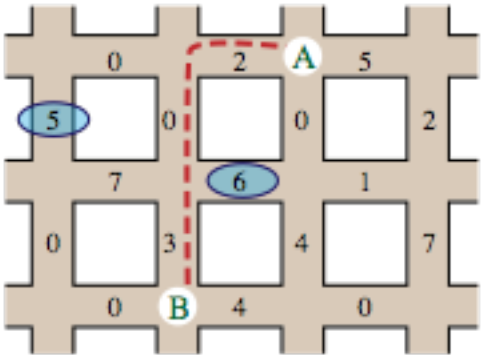
Debes manejar estos eventos implementando las rutinas `init()`, `changeH()`, `changeV()` y `escape()`, tal cómo se describen a continuación.

Ejemplos



La figura de arriba representa un plano inicial con `R = 3` calles horizontales y `C = 4` calles verticales, y en cada segmento se indica el número inicial de wombats. Considera la siguiente secuencia de eventos:

- Una persona llega a la intersección `A = (0, 2)` y quiere escapar hacia la intersección `B = (2, 1)`. El menor número posible de wombats con el que se puede cruzar es `2`, siguiendo la ruta indicada por la línea discontinua.
- Otra persona llega a la intersección `X = (0, 3)` y quiere escapar hacia la intersección `Y = (2, 3)`. El menor número de wombats con el que se puede cruzar es `7`, siguiendo de nuevo la ruta indicada por la línea discontinua.
- A continuación suceden dos eventos de tipo change: el número de wombats del segmento de arriba perteneciente a la calle vertical `0` cambia a `5`, y el número de wombats en el segmento de en medio perteneciente a la calle horizontal `1` cambia a `6`. Observa los números marcados en la figura que aparece a continuación.



- • Una tercera persona llega a la intersección `A = (0, 2)` y desea escapar hacia la intersección `B = (2, 1)`. Ahora el menor número de wombats que deberá pasar es `5`,

tal como está indicado por la nueva línea de trazos.

Implementación

Se te pide que envíes un archivo que implemente los procedimientos `init()`, `changeH()` y `changeV()` y la función `escape()`, de la siguiente forma:

Tu 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 te proporciona la distribución inicial del plano, y te permite inicializar variables globales y estructuras de datos. Sólo se llamará una 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]` indica el número de wombats que hay en el segmento situado entre las intersecciones `(P, Q)` y `(P, Q + 1)`.
- `V`: Un arreglo bidimensional de tamaño `(R - 1) × C`, donde `V[P][Q]` indica el número de wombats que hay en el segmento situado entre las intersecciones `(P, Q)` y `(P - 1, Q)`.

Tu Procedimiento: `changeH()`

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

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

Descripción

Se llamará a este procedimiento cuando cambie el número de wombats en el segmento horizontal situado entre las intersecciones `(P,Q)` y `(P, Q + 1)`.

Parámetros

- `P` : Indica la calle horizontal afectada ($0 \le P \le R - 1$).
- `Q` : Indica entre qué dos calles verticales está situado el segmento ($0 \le Q \le C - 2$).
- `W` : El nuevo número de wombats en este segmento ($0 \le W \le 1.000$).

Tu Procedimiento: `changeV()`

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

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

Descripción

Se llamará a este procedimiento cuando cambie el número de wombats en el segmento vertical situado entre las intersecciones (P,Q) y $(P + 1, Q)$.

Parámetros

- `P` : Indica entre que dos calles horizontales está el segmento ($0 \le P \le R - 2$).
- `Q` : Indica cual es la calle vertical afectada ($0 \le Q \le C - 1$).
- `W` : El nuevo número de wombats en este segmento de calle ($0 \le W \le 1.000$).

Tu Función: `escape()`

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

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

Descripción

Esta función tiene que calcular el mínimo número de wombats que una persona debe pasar cuando se mueve de la intersección $(0, V1)$ a $(R-1, V2)$.

Parámetros

- `V1` : Indica donde empieza la persona en la fila horizontal 0 ($0 \le V1 \le C-1$).
- `V2` : Indica donde acaba la persona en la fila horizontal $R-1$ ($0 \le V2 \le C-1$).
- *Returns*: El mínimo número de wombats que la persona debe pasar.

Ejemplo de Sesión

La siguiente entrada 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

- Límite de tiempo: 15 seconds
- Límite de memoria : 256 MB
- $2 \le R \le 5.000$
- $1 \le C \le 200$
- A lo sumo 500 cambios (llamadas ya sea a `changeH()` o `changeV()`)
- A lo sumo 200.000 llamadas a `escape()`
- A lo sumo 1.000 wombats sobre cualquier segmento en cualquier momento

Subtareas

Subtask	Points	Additional Input Constraints
1	9	$C = 1$
2	12	$R,C \le 20$, y no habrá llamadas a `changeH()` ni a `changeV()`
3	16	$R,C \le 100$, y habrá a lo sumo 100 llamadas a `escape()`
4	18	$C = 2$
5	21	$C \le 100$
6	24	*(Ninguna)*

Experimentación

El calificador de muestra instalado en su computador leerá la entrada a partir del archivo `wombats.in` ,que deberá estar estructurado siguiendo el siguiente formato:

- 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`
- siguientes `E` líneas: un evento por línea, en el order en el cual los eventos occurren

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 para cada evento tiene que tener uno de los formatos siguientes:

- 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`

El ejemplo desarrollado en el texto se tendría que dar 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 referidas a los Lenguajes

C/C++	Tienes que agregar `#include "wombats.h"` . Tienes que definir la `unit`
Pascal	`Wombats` . Todos los arreglos están numerados desde `0` (no desde `1`).

Mira las solution templates en tu ordenador para encontrar ejemplos.