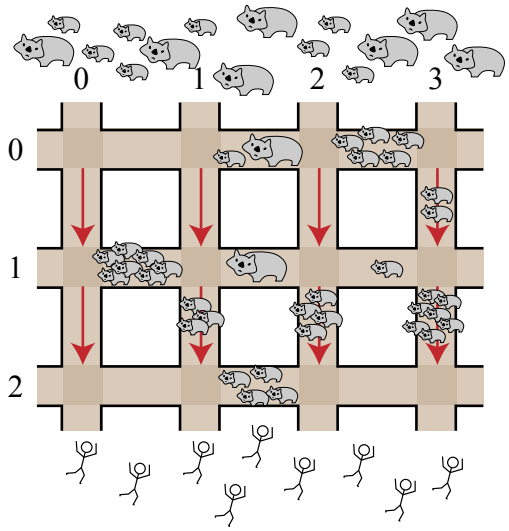
International Olympiad in Informatics 2013
6-13 July 2013
Brisbane, Australia

wombats

Português — 1.0

A cidade de Brisbane foi tomada de assalto por enormes vombates gigantes, e você tem de guiar as pessoas para um local seguro (um vombate é um animal originário da Austrália).

As ruas de Brisbane estão dispostas num enorme grade/grelha. Existem `R` ruas horizontais no sentido este-oeste, numeradas `0, ..., (R - 1)` por ordem de norte para sul, e `C` ruas verticais no sentido norte-sul, numeradas `0, ..., (C - 1)` por ordem de oeste para este, como é ilustrado na figura em baixo.



Os vombates invadiram a partir do norte, e as pessoas estão a escapar para sul. As pessoas podem correr ao longo de ruas horizontais em qualquer direção, mas nas ruas verticais *apenas podem correr para sul*, em direção à segurança.

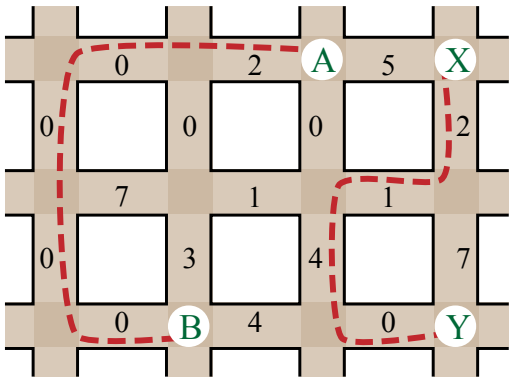
A interseção da rua horizontal `P` com a rua vertical `Q` é denotada por `(P, Q)`. Cada segmento de rua entre duas interseções contém um certo número de vombates, e estes números podem mudar ao longo do tempo. A sua tarefa é guiar cada pessoa de uma dada intersecção a norte (na rua horizontal `0`) para uma dada intersecção a sul (na rua horizontal `R - 1`), levando-a numa rota que passa pelo mínimo número possível de vombates.

Para começar, ser-lhe-á dado o tamanho da grade/grelha e o número de vombates de cada segmento de rua. A seguir a isto ser-lhe-ão dados uma série de `E` eventos, cada um dos quais é:

- uma *mudança*, que altera o número de vombates num certo segmento de rua; ou
- uma *fuga*, onde uma certa pessoa chega a uma dada intersecão na rua horizontal `0`, e você tem de descobrir uma rota para uma dada interseção na rua horizontal `R - 1` que passe pelo menor número possível de vombates.

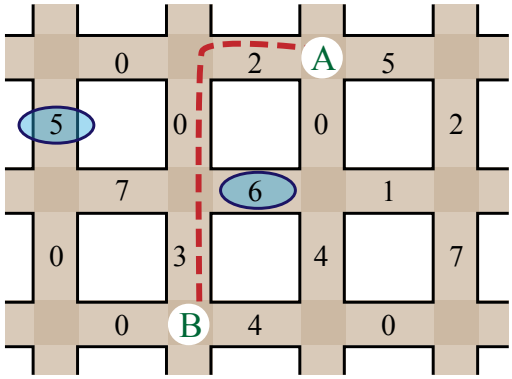
Você tem de lidar com estes eventos implementando as rotinas `init()`, `changeH()`, `changeV()` e `escape()`, como é descrito a seguir.

Exemplos



A figura de cima mostra um mapa inicial com `R = 3` ruas horizontais e `C = 4` ruas verticais, com o número de vombates marcado em cada segmento. Considere a seguinte série de eventos:

- Uma pessoa chega à interseção `A = (0, 2)` e deseja escapar para a interseção `B = (2, 1)`. O menor número de vombates por onde pode passar é `2`, como indicado pela linha a tracejado.
- Outra pessoa chega à intersecção `X = (0, 3)` e deseja escapar para a interseção `Y = (2, 3)`. O menor número de vombates por onde pode passar é `7`, novamente indicado pela linha a tracejado.
- Dois eventos de mudança ocorrem: o número de vombates no segmento do topo da rua vertical `0` muda para `5`, e o número de vombates no segmento do meio da rua horizontal `1` muda para `6`. Veja os números circundados na figura seguinte.



- Uma terceira pessoa chega à interseção `A = (0, 2)` e deseja escapar para a interseção `B = (2, 1)`. Agora, o menor número de vombates em que ela pode passar é `5`, como indicado pela nova linha tracejada.

Implementação

Você deve submeter um ficheiro implementando os procedimentos `init()`, `changeH()` e `changeV()` e a função `escape()`, tal como se segue:

O seu procedimento: `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);
```

Descrição

Este procedimento dá-lhe a configuração inicial do mapa, e permite-lhe inicializar quaisquer estruturas de dados e variáveis globais. Será chamado apenas uma vez, antes de qualquer chamada a `changeH()`, `changeV()` ou `escape()`.

Parâmetros

- `R`: O número de ruas horizontais.
- `C`: O número de ruas verticais.
- `H`: Uma matriz de tamanho `R × (C - 1)`, onde `H[P][Q]` dá o número de vombates no segmento de rua horizontal entre as interseções `(P, Q)` e `(P, Q + 1)`.
- `V`: Uma matriz de tamanho `(R - 1) × C`, onde `V[P][Q]` dá o número de vombates no segmento de rua vertical entre as interseções `(P, Q)` e `(P + 1, Q)`

O seu procedimento: `changeH()`

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

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

Descrição

Este procedimento será chamado quando se altera o número de vombates no segmento de rua horizontal entre as intersecões (P, Q) e $(P, Q + 1)$.

Parâmetros

- `P`: Indica qual rua horizontal é afetada ($0 \le P \le R - 1$).
- `Q`: Indica entre que duas ruas verticais o segmento está ($0 \le Q \le C - 2$).
- `W`: O novo número de vombates neste segmento de rua ($0 \le W \le 1{,}000$).

O seu procedimento: `changeV()`

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

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

Descrição

Este procedimento será chamado quando se altera o número de vombates no segmento de rua vertical entre as interseções (P, Q) e $(P + 1, Q)$.

Parâmetros

- `P`: Indica entre que duas ruas horizontais o segmento está ($0 \le P \le R - 2$)
- `Q`: Indica qual rua vertical é afetada ($0 \le Q \le C - 1$).
- `W`: O novo número de vombates neste segmento de rua ($0 \le W \le 1{,}000$).

A sua função: `escape()`

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

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

Descrição

Esta função deve calcular o menor número possível de vombates que uma pessoa tem de atravessar quando viaja da interseção $(0, V1)$ para $(R-1, V2)$.

Parâmetros

- `V1`: Indica onde a pessoa começa na linha horizontal 0 ($0 \le V1 \le C-1$).
- `V2`: Indica onde a pessoa termina na linha horizontal $R-1$ ($0 \le V2 \le C-1$).
- *Retorno*: O menor número possível de vombates que a pessoa tem de atravessar.

Sessão de exemplo

A seguinte sessão descreve o exemplo 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`

Restrições

- Limite de tempo: 15 seconds
- Limite de memória: 256 MiB
- $2 \le R \le 5{,}000$
- $1 \le C \le 200$
- No máximo 500 mudanças (chamadas a `changeH()` ou `changeV()`)
- No máximo 200,000 chamadas a `escape()`
- No máximo 1,000 vombates num dado segmento em qualquer altura

Sub-Tarefas

Sub-Tarefa	Pontos	Restrições adicionais
1	9	$C = 1$
2	12	$R,C \leq 20$, e não haverá chamadas a `changeH()` ou `changeV()`
3	16	$R,C \leq 100$, e existirão no máximo 100 chamadas a `escape()`
4	18	$C = 2$
5	21	$C \leq 100$
6	24	*(Nenhuma)*

Experimentação

O avaliador exemplo no seu computador irá ler o input do ficheiro `wombats.in`, que deverá estar no seguinte formato:

- linha 1: `R C`
- linha 2: `H[0][0] … H[0][C-2]`
- …
- linha $(R + 1)$: `H[R-1][0] … H[R-1][C-2]`
- linha $(R + 2)$: `V[0][0] … V[0][C-1]`
- …
- linha $(2R)$: `V[R-2][0] … V[R-2][C-1]`
- linha seguinte: `E`
- seguintes `E` linhas: um evento por linha, na ordem em que os eventos ocorrem

Se $C = 1$, as linhas vazias contendo o número de vombates nas ruas horizontais (linhas `2` até $R + 1$) não são necessárias.

A linha para cada evento deve estar num dos seguintes 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`

O exemplo anterior deve ser dado no seguinte 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 sobre as linguagens

C/C++	Você deve incluir `#include "wombats.h"`.
Pascal	Você deve definir uma `unit Wombats`. Todos os vetores são numerados começando por `0` (e não `1`).

Veja os templates de solução na sua máquina para exemplos.