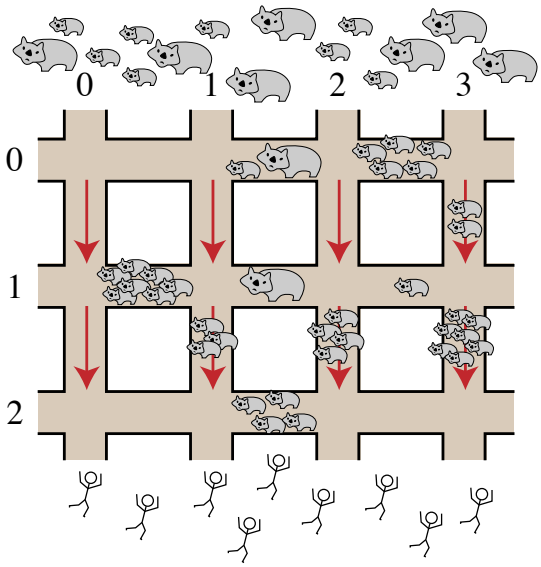
International Olympiad in Informatics
2013
6-13 July 2013
Brisbane, Australia

wombats

Japanese — 1.1

ブリスベン市は突然変異した大型のウォンバットたち (wombats) に乗っ取られてしまったので，あなたは人々を避難所まで導かなければならない．

ブリスベン市の道路は大きなグリッド状に並んでいる．以下の図のように，東西に水平に伸びる R 本の道路 (北から南に $0, \cdots, (R-1)$ の順に番号が付けられている) と，南北に垂直に伸びる C 本の道路 (西から東に $0, \cdots, (C-1)$ の順に番号が付けられている) がある．



ウォンバットたちは北から侵略し，人々は南へ逃げているところである．人々は水平方向の道路に沿って双方向に走ることができるが，垂直方向の道路に沿っては**南方向へだけ**，つまり避難所に向かう方向にだけ走ることができる．

水平方向の道路 P と垂直方向の道路 Q との交差点を (P, Q) と表記する．2 個の交差点を結ぶ道路の辺 (segment) には何匹かのウォンバットがいて，その数は時間の経過によって変化するかもしれない．あなたの課題は，通過しなければならないウォンバットの数が最小になるように，それぞれの人を北側 (水平方向の道路 0 上) のある交差点から，南側 (水平方向の道路 $R-1$ 上) のある交差点へ連れて行くことである．

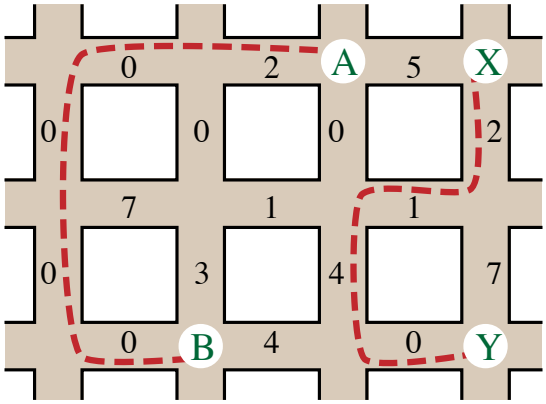
はじめに，グリッドの大きさと各辺にいるウォンバットの数が与えられる．それに続いて，E 個のイベントが与えられる．イベントはそれぞれ以下のうちのいずれかである：

- **変化** (change) イベントでは，ある辺上のウォンバットの数が変化する．

- **避難** (escape) イベントでは，ある人が水平方向の道路 `0` 上の与えられた交差点に到着するので，あなたは水平方向の道路 `R-1` 上の与えられた交差点へ，通過しなければならないウォンバットの数が最小になる経路を見つけなければならない．

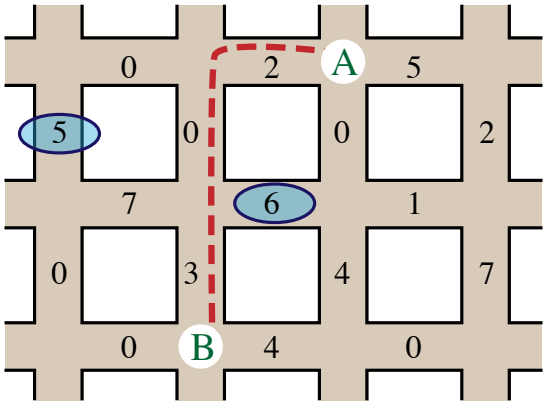
これらのイベントを処理するために，ルーチン `init()`，`changeH()`，`changeV()`，`escape()` を以下で述べる通りに実装しなければならない．

例 (Examples)



この図は `R = 3` 本の水平方向の道路と `C = 4` 本の垂直方向の道路をもつ初期状態の地図である．各辺上にウォンバットの数が示されている．次のようなイベントが順に来た場合を考える：

- 交差点 `A = (0, 2)` に人が到着し，交差点 `B = (2, 1)` へ逃げようとしている．彼女が通過しなければならないウォンバットの数は最小で `2` であり，その経路は図中に破線で表示されている．
- 交差点 `X = (0, 3)` に新たに人が到着し，交差点 `Y = (2, 3)` へ逃げようとしている．彼が通過しなければならないウォンバットの数は最小で `7` であり，その経路は図中に破線で表示されている．
- 2 個の変化イベントが発生する： 垂直方向の道路 `0` の一番上の辺にいるウォンバットの数が `5` に変化する．また，水平方向の道路 `1` の中央の辺にいるウォンバットの数が `6` に変化する．下の図の丸で囲まれた部分を見よ．



- A = (0, 2) に 3 人目の人が到着し，交差点 B = (2, 1) へ逃げようとしている．彼女が通過しなければならないウォンバットの数は今度は最小で 5 である．その経路は図中に新しく破線で表示されている．

実装 (Implementation)

あなたは，次のようなプロシージャー `init()`，`changeH()`，`changeV()` と関数 `escape()` を実装し，それらを含む 1 つのファイルを提出しなければならない．

あなたのプロシージャー (Your Procedure): `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);
```

説明 (Description)

このプロシージャーでは地図の初期配置が与えられ，あなたは任意のグローバル変数とデータ構造を初期化することができる．このプロシージャーは，`changeH()`，`changeV()`，`escape()` のどの呼び出しよりも前に 1 回だけ呼び出される．

引数 (Parameters)

- `R`：水平方向の道路の本数
- `C`：垂直方向の道路の本数
- `H`：水平方向の辺にいるウォンバットの数を表す大きさ R × (C - 1) の 2 次元配列．H[P][Q] は交差点 (P, Q) と交差点 (P, Q + 1) を結ぶ水平方向の辺にいるウォンバットの数を表す．

- `V`：垂直方向の辺にいるウォンバットの数を表す大きさ `(R - 1) × C` の 2 次元配列．`V[P][Q]` は交差点 `(P, Q)` と交差点 `(P + 1, Q)` を結ぶ垂直方向の辺にいるウォンバットの数を表す．

あなたのプロシージャー (Your Procedure): `changeH()`

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

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

説明 (Description)

このプロシージャーは交差点 `(P, Q)` と `(P, Q + 1)` を結ぶ水平方向の辺上のウォンバットの数が変化したときに呼び出される．

引数 (Parameters)

- `P`：どの水平方向の道路が影響を受けるかを表す (`0 ≤ P ≤ R - 1`)．
- `Q`：変化する辺がどの 2 本の垂直方向の道路の間にあるかを表す (`0 ≤ Q ≤ C - 2`)．
- `W`：この辺上のウォンバットの新しい数 (`0 ≤ W ≤ 1,000`)．

あなたのプロシージャー (Your Procedure): `changeV()`

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

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

説明 (Description)

このプロシージャーは交差点 `(P, Q)` と `(P + 1, Q)` を結ぶ垂直方向の辺上のウォンバットの数が変化したときに呼び出される．

引数 (Parameters)

- `P`：変化する辺がどの 2 本の水平方向の道路の間にあるかを表す (`0 ≤ P ≤ R - 2`)．
- `Q`：どの垂直方向の道路が影響を受けるかを表す (`0 ≤ Q ≤ C - 1`)．
- `W`：この辺上のウォンバットの新しい数 (`0 ≤ W ≤ 1,000`)．

あなたの関数 (Your Function): `escape()`

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

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

説明 (Description)

この関数は交差点 $(0, V1)$ から $(R-1, V2)$ へ逃げるときに通過しなければならないウォンバットの数の最小値を計算しなければならない.

引数と戻り値 (Parameters)

- $V1$: 水平方向の道路 0 上のどこから逃げ始めるかを表す ($0 \le V1 \le C-1$).
- $V2$: 水平方向の道路 $R-1$ 上のどこへ逃げるかを表す ($0 \le V2 \le C-1$).
- **Returns** (戻り値): 逃げるときに通過しなければならないウォンバットの数の最小値.

やりとりの例 (Sample Session)

次のやりとりは，上に述べた例を表す.

関数呼び出し	戻り値
`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`

制限 (Constraints)

- 時間制限：20 秒
- メモリ制限：256 MiB
- $2 \le R \le 5{,}000$
- $1 \le C \le 200$

- 変化イベント (`changeH()` または `changeV()` の呼び出し) は最大 500 回
- `escape()` の呼び出しは最大 200,000 回
- すべての時点でのすべての辺上において，ウォンバットの数は 1,000 以下

小課題 (Subtasks)

小課題	得点	入力に関する追加の制約
1	9	C = 1
2	12	R, C ≤ 20 であり，`changeH()` も `changeV()` も呼び出されない
3	16	R, C ≤ 100 であり，`escape()` への呼び出し回数は最大で 100 回
4	18	C = 2
5	21	C ≤ 100
6	24	(なし)

試行 (Experimentation)

与えられる採点プログラムのサンプルは，ファイル `wombats.in` から入力を読み込む．このファイルは次のフォーマットで書かれていなければならない：

- 1 行目：`R C`
- 2 行目：`H[0][0] … H[0][C-2]`
- …
- (R + 1) 行目：`H[R-1][0] … H[R-1][C-2]`
- (R + 2) 行目：`V[0][0] … V[0][C-1]`
- …
- (2R) 行目：`V[R-2][0] … V[R-2][C-1]`
- その次の行：`E`
- その次の `E` 行：イベントが起こる順番に，1 行に 1 つのイベントが書かれている

`C = 1` のときは，水平方向の道路にいるウォンバットの数を書くための空行 (`2` 行目から `R + 1` 行目) はなくてもかまわない．

それぞれのイベントを表す行は，次のようなフォーマットに従わなければならない：

- `changeH(P, Q, W)` を表すには，`1 P Q W` とする．
- `changeV(P, Q, W)` を表すには，`2 P Q W` とする．
- `escape(V1, V2)` を表すには，`3 V1 V2` とする．

例えば，上に述べた例は次のフォーマットで与えられなければならない：

```
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
```

言語に関する注意 (Language Notes)

C/C++ `#include "wombats.h"` を行うこと．

Pascal `unit Wombats` を定義すること．すべての配列は `0` から始まる (`1` からではない)．

実装例については，あなたの計算機上に与えられている解法テンプレートを参照すること．