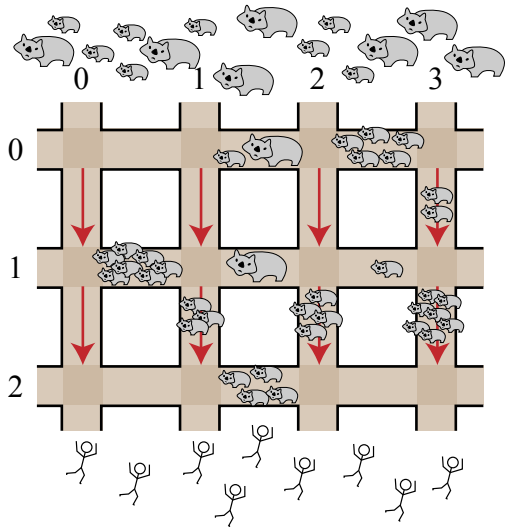
International Olympiad in Informatics 2013
6-13 July 2013
Brisbane, Australia

wombats

Netherlands — 1.1

Brisbane is overgenomen door grote gemuteerde wombats. Je moet de inwoners in veiligheid brengen.

De wegen in Brisbane liggen in een grid. Er zijn `R` horizontale wegen die van oost naar west lopen. Deze zijn van noord naar zuid als volgt genummerd: `0, ..., (R - 1)` en `C` verticale wegen die van noord naar zuid lopen. Deze zijn van west naar oost als volgt genummerd: `0, ..., (C - 1)` zoals je in het volgende plaatje kunt zien:



De wombats vallen vanuit het noorden binnen. De inwoners ontsnappen naar het zuiden. De inwoners kunnen op horizontale wegen beide kanten op rennen, maar op verticale wegen rennen ze altijd naar het zuiden, richting veiligheid.

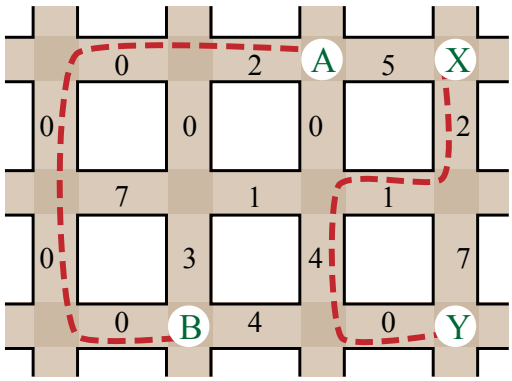
Het kruispunt van de horizontale weg `P` met de verticale weg `Q` wordt aangegeven als `(P, Q)` . Op elk stuk weg tussen twee kruispunten kan een aantal wombats staan. Dit aantal wombats kan veranderen in de tijd. Jouw taak is het om elke persoon vanaf een gegeven kruispunt in het noorden (op horizontale weg 0) te leiden naar een voorgeschreven kruispunt in het zuiden (op horizontale weg `R - 1`), over een route die langs zo min mogelijk wombats leidt.

Als eerste krijg je het formaat van de grid en het aantal wombats op elk stukje weg. Hierna krijg je een lijst van `E` gebeurtenissen. Elke gebeurtenis is ofwel:

- een *change* (wijziging), waarbij het aantal wombats op een stuk weg verandert; of
- een *escape* (ontsnapping), waarbij een persoon aankomt op een kruispunt op horizontale weg `0`, en je een route moet vinden naar een gegeven kruispunt op horizontale weg `R - 1` dat langs zo weinig mogelijk wombats leidt.

Je moet deze gebeurtenissen afhandelen door de volgende routines te implementeren: `init()`, `changeH()`, `changeV()` en `escape()`, zoals hieronder beschreven.

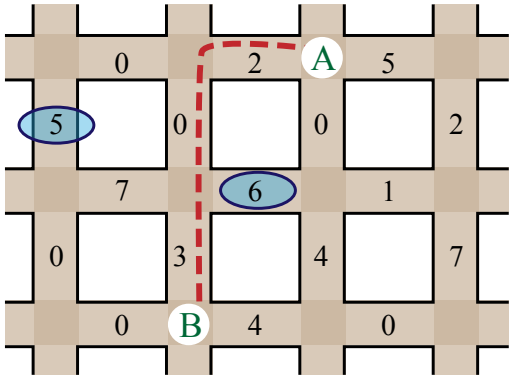
Voorbeelden



In het plaatje hierboven zie je de kaart aan het begin waarbij er `R = 3` horizontale wegen en `C = 4` verticale wegen zijn. Het aantal wombats op elk stukje weg is ook aangegeven.

Ga uit van de volgende serie gebeurtenissen:

- Een persoon komt aan op kruispunt `A = (0, 2)` en wil ontsnappen naar kruispunt `B = (2, 1)`. Het minimale aantal wombats waar deze persoon langs komt is `2`, zoals ook aangegeven met een stippellijn.
- Een tweede persoon komt aan op kruispunt `X = (0, 3)` en wil ontsnappen naar kruispunt `Y = (2, 3)`. Het minimale aantal wombats waar deze persoon langs moet is `7`, ook aangegeven met een stippellijn.
- Er vinden twee veranderingen plaats. Het aantal wombats op het bovenste stuk van de verticale weg `0` verandert naar `5`, en het aantal wombats op het middelste stuk van de horizontale weg `1` wordt `6`. Kijk ook naar de omcirkelde getallen in het plaatje hieronder.



- Een derde persoon komt nu aan op kruispunt `A = (0, 2)` en wil ontsnappen naar kruispunt `B = (2, 1)`. Het minimale aantal wombats waar deze persoon langs moet is `5`, zoals aangegeven met de nieuwe stippellijn.

Implementatie

Je moet een bestand insturen dat de procedures `init()`, `changeH()` and `changeV()` en de functie `escape()` als volgt implementeert:

Jouw 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);
```

Beschrijving

Deze procedure geeft je de oorspronkelijke indeling van de kaart. Je kunt nu ook je globale variabelen en datastructuren initialiseren. Deze functie wordt precies één keer aangeroepen, voor elke aanroep van `changeH()`, `changeV()` of `escape()`.

Parameters

- `R`: Het aantal horizontale wegen.
- `C`: Het aantal verticale wegen.
- `H`: Een twee-dimensionale array van grootte `R × (C - 1)`, waarbij `H[P][Q]` het aantal wombats aangeeft op het stuk horizontale weg tussen de kruispunten `(P, Q)` en `(P, Q + 1)`.
- `V`: Een twee-dimensionale array van grootte `(R - 1) × C`, waarbij `V[P][Q]` het aantal wombats aangeeft op het verticale stuk weg tussen de kruispunten `(P, Q)` en `(P + 1, Q)`.

Jouw procedure: changeH()

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

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

Beschrijving

Deze procedure wordt aangeroepen als het aantal wombats op het horizontale stuk weg tussen de kruispunten (P, Q) en $(P, Q + 1)$ verandert.

Parameters

- P : geeft aan welke horizontale weg dit betreft ($0 \le P \le R - 1$).
- Q : geeft aan tussen welke twee verticale wegen het stuk weg ligt ($0 \le Q \le C - 2$).
- W : het aantal nieuwe wombats op dit stuk weg ($0 \le W \le 1.000$).

Jouw procedure: `changeV()`

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

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

Beschrijving

Deze procedure wordt aangeroepen als het aantal wombats op het verticale stuk weg tussen de kruispunten (P, Q) en $(P + 1, Q)$ verandert.

Parameters

- `P` : geeft aan tussen welke twee horizontale wegen het stuk weg ligt ($0 \le P \le R - 2$).
- `Q` : geeft aan welke verticale weg dit betreft ($0 \le Q \le C - 1$).
- `W` : het aantal nieuwe wombats op dit stuk weg ($0 \le W \le 1.000$).

Jouw functie: `escape()`

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

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

Beschrijving

Deze functie berekent het minimale aantal wombats waar een persoon langs moet om van kruispunt $(0, V1)$ naar $(R-1, V2)$ te gaan.

Parameters

- `V1` : geeft aan waar een inwoner begint op horizontale weg 0 ($0 \le V1 \le C-1$).
- `V2` : geeft aan waar een inwoner naar toe moet op rij horizontale weg `R-1` ($0 \le V2 \le C-1$).
- *Antwoord*: Het minimale aantal wombats waar de inwoner langs moet.

Voorbeeld

Hieronder wordt het voorbeeld van hierboven uitgewerkt:

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`

Randvoorwaarden

- Tijdslimiet: 20 seconden
- Geheugenlimiet: 256 Mb
- $2 \leq R \leq 5.000$
- $1 \leq C \leq 200$
- Maximaal 500 wijzigingen (aanroepen van `changeH()` of `changeV()`)
- Maximaal 200.000 aanroepen van `escape()`
- Op elk tijdstip zijn er maximaal 1.000 wombats op een stuk weg.

Subtasks

Subtask	Punten	Aanvullende randvoorwaarden
1	9	$C = 1$
2	12	$R,C \le 20$, en geen aanroepen van `changeH()` of `changeV()`
3	16	$R,C \le 100$, en maximaal 100 aanroepen van `escape()`
4	18	$C = 2$
5	21	$C \le 100$
6	24	*(Geen)*

Stoeien

De voorbeeld grader op je computer leest de invoer uit `wombats.in`. Dit bestand is in het volgende formaat;

- regel 1: `R C`
- regel 2: `H[0][0] … H[0][C-2]`
- …
- regel $(R + 1)$: `H[R-1][0] … H[R-1][C-2]`
- regel $(R + 2)$: `V[0][0] … V[0][C-1]`
- …
- regel $(2R)$: `V[R-2][0] … V[R-2][C-1]`
- volgende regel: `E`
- volgende `E` regels: een gebeurtenis per regel, op de volgorde van de gebeurtenissen.

Als $C = 1$, dan zijn de lege regels met het aantal wombats op de horizontale wegen (regels `2` tot en met $R + 1$) niet noodzakelijk.

Elke regel die een gebeurtenis beschrijft moet een van de volgende formaten hebben:

- om een aanroep van `changeH(P, Q, W)` aan te geven: `1 P Q W`
- om een aanroep van `changeV(P, Q, W)` aan te geven: `2 P Q W`
- om een aanroep van `escape(V1, V2)` aan te geven: `3 V1 V2`

Het voorbeeld hierboven wordt in het volgende formaat aangegeven:

```
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
```

Opmerkingen bij de talen

C/C++	You must `#include "wombats.h"`.
Pascal	You must define the `unit Wombats`. All arrays are numbered beginning at `0` (not `1`).

See the solution templates on your machine for examples.