巴勒與夏拉正在玩一個遊戲。遊戲盤是一個由 R 列（以 0, …, $R-1$ 標示）乘以 C 行（以 0, …, $C-1$ 標示）單元（cells）所構成的網格（grid）。我們以 (P, Q) 來表示位於第 P 列、第 Q 行的單元。每個單元包含一個非負整數，遊戲一開始時所有單元的值皆初始化為 0。

遊戲進行的方式如下：在任何時間點，巴勒可以

- 藉由指定一個單元 (P, Q) 所包含的整數值來進行更新（update）。
- 要求夏拉計算（calculate）一個矩形區域裡全部單元所包含數值的最大公因數（greatest common divisor，GCD），這個矩形透過給定其兩個對角 (P, Q) 與 (U, V) 來定義（對角單元包含在矩形內）。

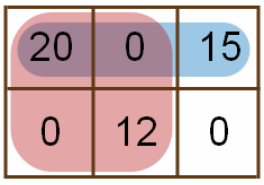
巴勒在玩膩這個遊戲而跑到外面玩板球（cricket）前，最多將進行 $N_U + N_Q$ 個動作（更新 N_U 次及要求計算 N_Q 次）。

你的任務是算出正確的答案。

範例（Examples）

假設 $R = 2$ 而 $C = 3$，並且巴勒一開始進行以下的更新：

- 更新單元 $(0, 0)$ 的數值為 20；
- 更新單元 $(0, 2)$ 的數值為 15；
- 更新單元 $(1, 1)$ 的數值為 12。

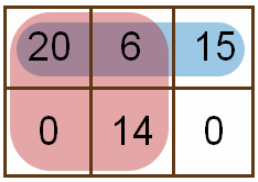


網格的結果如上圖所示。巴勒接著可能會問以下矩形區域裡，全部單元所包含數值的最大公因數：

- 對角 $(0, 0)$ 與 $(0, 2)$：在這個矩形裡的三個整數分別為 20、0，以及 15，所以它們的最大公因數是 5。
- 對角 $(0, 0)$ 與 $(1, 1)$：在這個矩形裡的四個整數分別為 20、0、0，以及 12，所以它們的最大公因數是 4。

現在假設巴勒做了以下的更新：

- 更新單元 $(0, 1)$ 的數值為 6；
- 更新單元 $(1, 1)$ 的數值為 14。



新的網格如上圖所示。巴勒接著可能又會問以下矩形區域裡，全部單元所包含數值的最大公因數：

- 對角 $(0, 0)$ 與 $(0, 2)$：現在這個矩形裡的三個整數分別為 20、6，以及 15，所以它們的最大公因數是 1。
- 對角 $(0, 0)$ 與 $(1, 1)$：現在這個矩形裡的四個整數分別為 20、6、0，以及 14，所以它們的最大公因數是 2。

此處巴勒總共進行了 $N_U = 5$ 次更新與 $N_Q = 4$ 次要求計算。

程式實作（Implementation）

你應該提交（submit）一個檔案來實作程序（procedures）`init()`與 `update()`，以及函式（function）`calculate()`如下。

為了協助你，你的電腦裡提供了範本解答（`game.c`、`game.cpp`，以及 `game.pas`），每一個解答皆有包含函式 `gcd2(X, Y)`來計算兩個給定非負整數 X 與 Y 的最大公因數。若 $X = Y = 0$，則 `gcd2(X, Y)`也會回傳 0。

這個函式夠快能夠得到所有的分數；在最糟情況下，執行時間正比於 $\log(X + Y)$。

你的程序：`init()`

C/C++
```
void init(int R, int C);
```

Pascal
```
procedure init(R, C : LongInt);
```

敘述

你送出的內容必須實作這個程序。

這個程序會給定你網格的大小，並且允許你初始化任何全域變數與資料結構。在呼叫 `update()`或 `calculate()`之前必須先呼叫這個程序，並且只會呼叫一次。

參數

- `R`：列的數目

- C：行的數目

你的程序：update()

C/C++ `void update(int P, int Q, long long K);`

Pascal `procedure update(P, Q : LongInt; K: Int64);`

敘述

你送出的解答必須實作這個程序。

當巴勒更改某個網格單元的值時，這個程序將被呼叫。

參數

- P：網格單元所在的列（$0 \le P \le R-1$）。
- Q：網格單元所在的行（$0 \le Q \le C-1$）。
- K：這個網格單元的新數值（$0 \le K \le 10^{18}$）。可能跟目前的數值相同。

你的函式：calculate()

C/C++ `long long calculate(int P, int Q, int U, int V);`

Pascal `function calculate(P, Q, U, V : LongInt) : Int64;`

敘述

你送出的內容必須實作這個函式。

這個函式應該計算包含於對角 (P, Q) 與 (U, V) 這個矩形區域裡，全部單元所包含數值的最大公因數。這個矩形的範圍包含單元 (P, Q) 與 (U, V)。

如果這個矩形裡的所有整數都是 0，那麼這個函式也應該回傳 0。

參數

- P：這個矩形左上角單元所在的列（$0 \le P \le R-1$）。
- Q：這個矩形左上角單元所在的行（$0 \le Q \le C-1$）。
- U：這個矩形右下角單元所在的列（$P \le U \le R-1$）。
- V：這個矩形右下角單元所在的行（$Q \le V \le C-1$）。
- *Returns*：這個矩形區域裡，全部單元所包含數值的最大公因數；若所有數值皆為 0，那麼就回傳 0。

實例（Sample Session）

以下描述上述例子：

Function Call	Returns
`init(2, 3)`	
`update(0, 0, 20)`	
`update(0, 2, 15)`	
`update(1, 1, 12)`	
`calculate(0, 0, 0, 2)`	5
`calculate(0, 0, 1, 1)`	4
`update(0, 1, 6)`	
`update(1, 1, 14)`	
`calculate(0, 0, 0, 2)`	1
`calculate(0, 0, 1, 1)`	2

限制

- 時間限制：2 秒
- 記憶體限制：512 MiB
- $1 \le R, C \le 10^9$
- $0 \le K \le 10^{18}$，K 是巴勒放置於網格單元的任何整數值。

子任務 (詳細內容請參見英文版)

子任務	配分	R	C	N_U	N_Q	Time limit	Memory limit
1	10						
2	28						
3	26						
4	36						

實驗

在你電腦的範例評分機制（grader）將會由檔案 `game.in` 讀取輸入資訊。這個檔案格式如下：

- 第 1 行: `R C N`
- 接下來 *N* 行: 依照動作發生的順序，一個動作一行

表示動作的那一行必須是以下其中一種格式：

- 表示 `updaqte(P, Q, K)`：`1 P Q K`
- 表示 `calculate(P, Q, U, V)`：`2 P Q U V`

例如，上述的例子應該以下列的格式呈現：

```
2 3 9
1 0 0 20
1 0 2 15
1 1 1 12
2 0 0 0 2
2 0 0 1 1
1 0 1 6
1 1 1 14
2 0 0 0 2
2 0 0 1 1
```

語言註釋（Language Notes）

C/C++　你必須 `#include “game.h”`。

Pascal　你必須定義 `unit Game`。所有陣列由 0 開始標示（不是 1）。

因為網格單元的整數值可能非常大，C/C++的使用者建議使用型別 `long long`，Pascal 的使用者建議使用型別 `Int64`。