



International Olympiad in Informatics  
2013

6-13 July 2013

Brisbane, Australia

Day 2 tasks

game  
中文(澳門) —  
1.0

Bazza 和 Shazza 正在玩一個遊戲。遊戲在一塊網格板上進行，其中板上有  $R$  個橫行，其編號為  $0, \dots, R - 1$ ，和  $C$  個直列，其編號為  $0, \dots, C - 1$ 。我們說  $(p, q)$  為在第  $p$  橫行和第  $q$  直列上的網格。每個網格都包含一個非負的整數，並且在遊戲開始時，它們的值都是 0。

遊戲以如下方式進行。在任何時候，Bazza 都可以做以下二個動作：

- 更新(update)一個網格  $(P, Q)$ ，使它變成另一個整數。
- 要求 Shazza 去計算出在一些矩形內所有網格中的整數的最大公因數 (GCD)，其對角座標為  $(P, Q)$  和  $(U, V)$  (含)。

在Bazza感到沉悶而出去打板球之前，他將會做  $N_U + N_Q$  個動作（更新網格  $N_U$  次和問  $N_Q$  次問題）。

你的任務是計算出正確的答案。

## 例子

假設  $R = 2$  和  $C = 3$ ，Bazza 開始作以下的更新：

- 更新網格  $(0, 0)$  為 20；
- 更新網格  $(0, 2)$  為 15；
- 更新網格  $(1, 1)$  為 12。

20	0	15
0	12	0

最終結果如上圖所示。Bazza 然後詢問以下矩形內的最大公因數 (GCD)：

- 對角坐標為  $(0, 0)$  和  $(0, 2)$ ：這個矩形內的三個整數分別是 20, 0 和 15, 而他們的最大公因數 (GCD) 為 5。
- 對角坐標為  $(0, 0)$  和  $(1, 1)$ ：這個矩形內的四個整數分別是 20, 0, 0 和 12, 而他們的最大公因數 (GCD) 為 4。

現在假設 Bazza 作了以下的更新：

- 更新網格 (update)  $(0, 1)$  成為 6；
- 更新網格 (update)  $(1, 1)$  成為 14。

20	6	15
0	14	0

新的網格如上圖所示。Bazza 之後可以再次詢問以下矩形內的最大公因數 (GCD)：

- 對角坐標為  $(0, 0)$  和  $(0, 2)$ ：現在這個矩形內的三個整數分別為 20, 6 和 15, 而它們的最大公因數 GCD 為 1。
- 對角坐標為  $(0, 0)$  和  $(1, 1)$ ：現在這個矩形內的四個整數分別為 20, 6, 0 和 14, 而它們的最大公因數 GCD 為 2。

這裏 Bazza 作了  $N_u = 5$  個更新和  $N_q = 4$  次詢問。

## 程式實現

你應該提交一個編寫了子程序 (procedures) `init()` 和 `update()` 和函數 (function) `calculate()`，的檔案，其內容如下：

在你的電腦上我們提供了一些樣例題解模板來幫助你 (`game.c`, `game.cpp` 和 `game.pas`) 每個程序均包含了一個函數 (function) `gcd2(X, Y)` 去計算給定的二個非負整數  $X$  和  $Y$  的最大公因數 (GCD)。如果  $X = Y = 0$ ，則 `gcd2(X, Y)` 將會同樣地返回 `0`。

這個函數的速度是足夠可以得到滿分的；換言之，它在最差的情況下其執行的速度是與  $\log(X + Y)$  成正比的。

你要編寫的子程式：**init()**

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

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

描述

你提交的檔案必須要實現這個子程式。

這個子程式將提供給你網格的初始大小。它並可給你初始化所有全域變量的初始值。它只能被調用一次，並且要在調用 `update()` 或 `calculate()` 之前。

參數

- `R`：橫行的數目。
- `C`：直列的數目。

你要編寫的子程式：**update()**

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

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

描述

你提交的檔案必須要實現這個子程式。

當Bazza改變某個網格內的數值時，這個子程式就會被調用。

參數

- `P`：網格的橫行編號 ( $0 \leq P \leq R - 1$ )。
- `Q`：網格的直列編號 ( $0 \leq Q \leq C - 1$ )。
- `K`：網格內新的整數 ( $0 \leq K \leq 10^{18}$ )，這個數可能與原來的數字相同。

你的函數：**calculate()**

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

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

## 描述

你提交的檔案必須要實現這個子程式。

這個函數將計算出在一個對角坐標為  $(P, Q)$  及  $(U, V)$  的矩形內所有整數的最大公因數。 這個指定的矩形範圍是包括邊角（即網格  $(P, Q)$  及  $(U, V)$  在內）的。

如果這個矩形內的所有整數均為0，這個函數的返回值也應該為0。

## 參數

- $P$ ：矩形的左上角那格的橫行號碼 ( $0 \leq P \leq R - 1$ )。
- $Q$ ：矩形的左上角那格的直列號碼 ( $0 \leq Q \leq C - 1$ )。
- $U$ ：矩形的右下角那格的橫行號碼 ( $P \leq U \leq R - 1$ )。
- $V$ ：矩形的右下角那格的直列號碼 ( $Q \leq V \leq C - 1$ )。
- 返回值：所有在矩形內的整數的 GCD，但若所有這些整數均為 0 其返回值亦為 0。

## 樣例程序

以下的過程將描述上面的例子：

函數調用	返回
<code>init(2, 3)</code>	
<code>update(0, 0, 20)</code>	
<code>update(0, 2, 15)</code>	
<code>update(1, 1, 12)</code>	
<code>calculate(0, 0, 0, 2)</code>	5
<code>calculate(0, 0, 1, 1)</code>	4
<code>update(0, 1, 6)</code>	
<code>update(1, 1, 14)</code>	
<code>calculate(0, 0, 0, 2)</code>	1
<code>calculate(0, 0, 1, 1)</code>	2

## 限制條件

- 時間限制：參閱下列子任務。
- 記憶體限制：參閱下列子任務。
- $1 \leq R, C \leq 10^9$
- $0 \leq K \leq 10^{18}$ ，這裏的  $K$  是 Bazza 放在網格上的任意整數。

---

## 子任務

這部份請參閱英文版本。

這部份請參閱英文版本

---

## 程式實現

你電腦上的樣例 grader 將讀入文字檔 `game.in`，它的格式如下：

- 第 1 行：`R C N`
- 下  $N$  行：每行一個動作並根據每個動作發生的先後順序排列。

每一行動作資料必須是用以下格式表示：

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

例如：上面例子的資料將會是用以下格式的：

```
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
```

---

## 編譯語言注意事項

C/C++ 你必須包括 `#include "game.h"` 。

Pascal 你必須定義(define) `unit Game`。 All arrays are numbered beginning at `0` (not `1`)。

因為在網格上的整數將會很大，C/C++ 的使用者請使用 `long long` 類型，而 Pascal 的使用者請使用 `Int64` 類型。