CCO '18 P4 - Gradient Descent

Canadian Computing Olympiad: 2018 Day 2, Problem 1

Troy wants to play the following game with you.

He has a grid with R rows and C columns. Rows are numbered from 1 to R and columns are numbered from 1 to C. Let the cell at row p and column q be denoted as (p, q).

There are N tokens numbered from 1 to N. Token i is placed at (X_i, Y_i) where $1 \leq X_i \leq R$ and $1 \leq Y_i \leq C$. There may be multiple tokens at the same cell. In one second, Troy can move one token to a horizontally or vertically adjacent cell. The *score* of a cell is defined as the minimum number of seconds needed for Troy to move every token to this cell.

Your goal is to find the minimum score of any cell in the grid. Unfortunately, Troy does not tell you how many tokens there are or where they are placed. However, you may ask him questions. You can ask Troy to tell you the score of any cell (p, q). You may ask at most K questions before Troy gets bored.

Interaction Protocol

This problem is interactive: input will be given based on questions generated by your program.

First, read one line with three integers R, C, K $(1 \leq R, C \leq 10\,000\,000; 1 \leq K \leq 170)$.

After your program has read this line, your program may ask questions.

To ask a question about (p, q) $(1 \leq p \leq R; 1 \leq q \leq C)$, print one line in the format `? p q`. Then, read one line with one integer s $(0 \leq s \leq 2\,000\,000\,000)$, the score of (p, q).

Once your program determines the minimum score is Z, print one line in the format `! Z`. Your program must terminate immediately after printing this line.

The output must be flushed after every line is printed, including the last line. To flush you can use: `fflush(stdout)` or `cout << endl` in C/C++; `System.out.flush()` in Java; `flush(output)` in Pascal.

If any printed line is wrongly formatted or you ask more than K questions you will get an *incorrect* verdict.

For every test case, the grading system will have fixed integer values N, R, C, K, $X_1, \ldots, X_N$, $Y_1, \ldots, Y_N$ $(1 \leq N \leq 100; 1 \leq X_i \leq R; 1 \leq Y_i \leq C)$. These values will remain constant while your program is running. That is, the grading system is **not adaptive**.

For 5 of the 25 available marks, $R = 1$, $C \leq 90$ and $K = 90$.

For an additional 5 of the 25 available marks, $R = 1$, and $K = 90$.

For an additional 5 of the 25 available marks, $K = 170$.

For an additional 5 of the 25 available marks, $K = 100$.

For the remaining 5 marks, $K = 75$.

Sample Interaction 1

Request to grader	Feedback from Grader
	`1 10 90`
`? 1 3`	`9`
`? 1 7`	`11`
`? 1 4`	`8`
`! 8`	

Explanation of Output for Sample Input 1

This sample corresponds to tokens at cells $(1, 2)$, $(1, 4)$ and $(1, 10)$. It is guaranteed this sample will match the sample test in the grader.

The score of cell $(1, 3)$ is $1 + 1 + 7 = 9$.

The score of cell $(1, 7)$ is $5 + 3 + 3 = 11$.

The score of cell $(1, 4)$ is $2 + 0 + 6 = 8$ and this is the minimum in the grid.

For information, here are the scores in this example:

Column	1	2	3	4	5	6	7	8	9	10
Score	13	10	9	8	9	10	11	12	13	14

Sample Interaction 2

Request to grader	Feedback from Grader
	`5 4 170`
`? 2 4`	`11`
`? 1 4`	`15`
`? 3 3`	`7`
`! 7`	

Explanation of Output for Sample Input 2

This sample corresponds to tokens at cells $(2, 3)$, $(2, 3)$, $(4, 3)$ and $(5, 1)$.