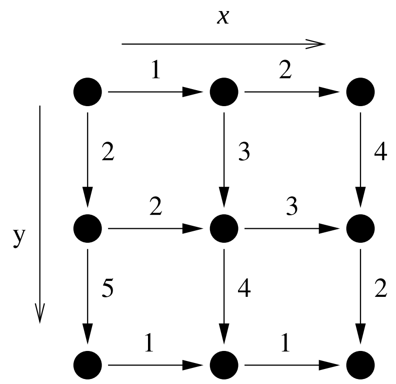
Input File: –
Output File: –
Source Code: `square.pas/.c/.cpp`

100 Points
Time Limit: 1 s
Memory Limit: 16 MB

Square

You are given a graph. All vertices are positioned on a square $N \times N$ grid N ($1 \leq N \leq 2003$). Each point of the grid is occupied by exactly one vertex. Each vertex has directed edges to its neighbor vertices to the right and to the bottom, if they exist. These edges are weighted with an integer w ($1 \leq w \leq 500\,000$). Every path starting at the upper left vertex (at $v_{1,1}$) to a vertex $v_{x,y}$ has the same weight. The weight of a path is the sum of the weight of the edges along the path.

In the following graph with $N = 3$, every path from vertex $v_{1,1}$ to $v_{2,2}$ has weight 4. The only path from $v_{1,1}$ to $v_{1,2}$ has weight 2.



You are given an integer L ($1 \leq L \leq 2\,000\,000\,000$). Your task is to search for a vertex $v_{x,y}$ so that a path from $v_{1,1}$ to $v_{x,y}$ has exactly weight L. The weights are not directly given to your program, and so it has to ask for a special weight by using a library. Your program is only allowed to ask for at most 6667 weights.

The library offers the following functions:
`getN()` and `getL()` return the values for N and L. `getWeight(x, y, direction)` returns the weight of the edge starting at $v_{x,y}$ to the right (`direction=0`) or to the bottom (`direction=1`).

If you have found a vertex $v_{x,y}$ with distance L, you have to call `solution(x, y)`. If no such vertex exists, call `solution(-1, -1)`. Your program will be terminated automatically after calling `solution(x, y)`. If you perform more then 6667 calls to `getWeight()` or your solution is incorrect, you will get no points for that testcase.

This task has neither input nor output files.

Library functions

C/C++
`int getN(void)`
`int getL(void)`
`int getWeight(int x, int y, int direction)`
`void solution(int x, int y)`
Pascal
`function getN: Longint`
`function getL: Longint`
`function getWeight(x, y, direction: Longint): Longint`
`procedure solution(x, y: Longint)`

You can find a sample implementation of the library in `~/ceoi` or `c:\ceoi`, but your program will be evaluated with another implementation.

For testing and debugging you can create the file `square.in`, which will be read by the sample library. The first line of `square.in` must contain the integers N and L. Each of the following N lines contains exactly $N-1$ integers, the weights of the horizontal edges. After that, the following $N-1$ lines contain N integers, the weights of the vertical edges.

The library is to be included via `#include "square_lib.h"` or `uses square_lib`.

Please note that the sample library you can use for testing is rather slow and memory consuming. This is because it has to read your input file into memory. Nevertheless you can assume that the evaluation library doesn't need any memory or running time at all.

Sample input for library

square.in	Protocol
3 4	getN() -> 3
1 2	getL() -> 4
2 3	getWeight(1,1,0) -> 1
1 1	getWeight(2,1,1) -> 3
2 3 4	solution(2,2)
5 4 2	