

# Turtle Mission: Robot and the Earthquake

Input file:            **standard input**  
Output file:           **standard output**  
Time limit:            3 seconds  
Memory limit:         256 megabytes

The world is a grid with  $n$  rows and  $m$  columns. The rows are numbered  $0, 1, \dots, n-1$ , while the columns are numbered  $0, 1, \dots, m-1$ . In this world, the columns are **cyclic** (i.e. the top and the bottom cells in each column are adjacent). The cell on the  $i$ -th row and the  $j$ -th column ( $0 \leq i < n, 0 \leq j < m$ ) is denoted as  $(i, j)$ .

**At time 0**, the cell  $(i, j)$  (where  $0 \leq i < n, 0 \leq j < m$ ) contains either a **rock** or **nothing**. The state of cell  $(i, j)$  can be described using the integer  $a_{i,j}$ :

- If  $a_{i,j} = 1$ , there is a rock at  $(i, j)$ .
- If  $a_{i,j} = 0$ , there is nothing at  $(i, j)$ .

As a result of aftershocks from the earthquake, the columns follow tectonic plate movements: each column moves cyclically **upwards** at a velocity of 1 cell per unit of time. Formally, for some  $0 \leq i < n, 0 \leq j < m$ , if  $(i, j)$  contains a rock at the moment, it will move from  $(i, j)$  to  $(i-1, j)$  (or to  $(n-1, j)$  if  $i = 0$ ).

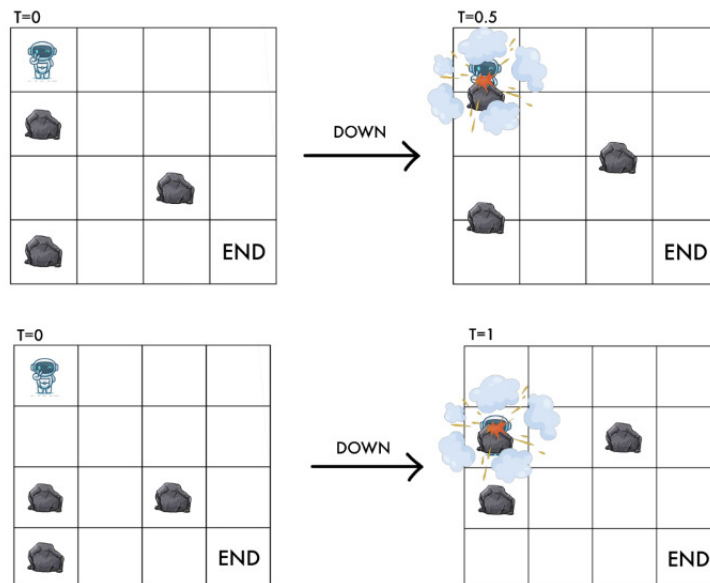
The robot called RT is initially positioned at  $(0, 0)$ . It has to go to  $(n-1, m-1)$  to carry out an earthquake rescue operation (to the bottom rightmost cell). The earthquake doesn't change the position of the robot, they only change the position of rocks in the world.

Let RT's current position be  $(x, y)$  ( $0 \leq x < n, 0 \leq y < m$ ), it can perform the following operations:

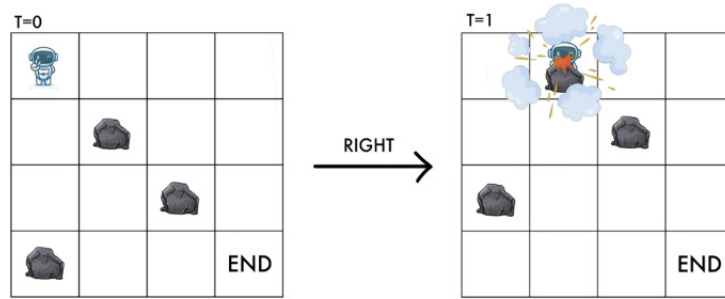
- Go one cell cyclically upwards, i.e. from  $(x, y)$  to  $((x+n-1) \bmod n, y)$  using 1 unit of time.
- Go one cell cyclically downwards, i.e.  $(x, y)$  to  $((x+1) \bmod n, y)$  using 1 unit of time.
- Go one cell to the right, i.e.  $(x, y)$  to  $(x, y+1)$  using 1 unit of time. (RT may perform this operation only if  $y < m-1$ .)

**Note that RT cannot go left using the operations nor can he stay at a position.**

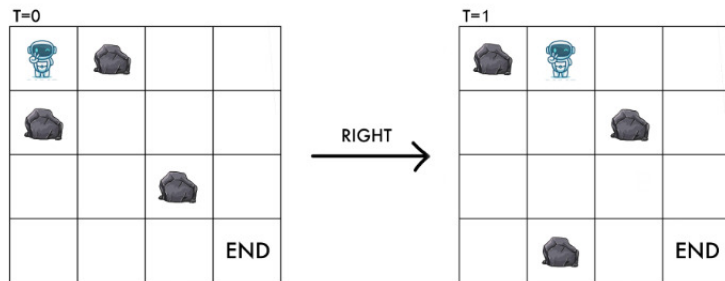
Unfortunately, RT will explode upon colliding with a rock. As such, when RT is at  $(x, y)$  and there is a rock at  $((x+1) \bmod n, y)$  or  $((x+2) \bmod n, y)$ , RT cannot move down or it will be hit by the rock.



Similarly, if  $y + 1 < m$  and there is a rock at  $((x + 1) \bmod n, y + 1)$ , RT cannot move right or it will be hit by the rock.



However, it is worth noting that if there is a rock at  $(x \bmod n, y + 1)$  and  $((x + 1) \bmod n, y)$ , RT can still move right safely.



Find the minimum amount of time RT needs to reach  $(n - 1, m - 1)$  without colliding with any rocks. If it is impossible to do so, output  $-1$ .

## Input

The first line of the input contains one integer  $t$  ( $1 \leq t \leq 10^4$ ) — the number of test cases.

In each test case, the first line contains two integers  $n, m$  ( $3 \leq n, m \leq 10^3$ ) — the size of the planet's boundaries.

Each of the next  $n$  lines contains  $m$  integers. The  $(j + 1)$ -th integer on the  $(i + 1)$ -th line ( $0 \leq i < n, 0 \leq j < m$ ) is  $a_{i,j}$  ( $0 \leq a_{i,j} \leq 1$ ), which denotes whether or not there is a rock at  $(i, j)$  at time 0.

**Additionally, it is guaranteed that  $a_{0,0} = 0$ , and  $a_{i,m-1} = 0$  for  $0 \leq i < n$ .** In other words, there is no rock at RT's initial position as well as column  $m - 1$ .

The sum of  $n \cdot m$  over all test cases does not exceed  $10^6$ .

## Output

For each test case:

- If the destination can be reached without colliding with any rocks, output a single integer — the minimum amount of time RT needs to reach  $(n - 1, m - 1)$ .
- Otherwise, output  $-1$ .







## Examples

standard input	standard output
6 4 5 0 1 0 0 0 0 0 1 0 0 1 0 1 1 0 0 0 0 0 0 3 3 0 0 0 1 0 0 0 0 0 5 3 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 3 7 0 0 1 0 0 1 0 1 0 1 0 1 0 0 0 1 0 0 0 0 0 3 4 0 1 0 0 1 0 0 0 0 1 1 0 5 5 0 0 0 0 0 0 1 0 1 0 0 1 0 1 0 0 1 0 1 0 0 0 0 1 0	7 3 3 8 -1 12
6 3 3 0 0 0 0 0 0 0 0 0 4 3 0 1 0 1 0 0 0 1 0 1 0 0 4 3 0 1 0 0 1 0 0 1 0 0 1 0 3 3 0 0 0 1 1 0 0 0 0 3 3 0 1 0 0 0 0	3 3 -1 -1 3 8
0 1 0 5 5 0 0 0 0 0	








## Note

Visual explanation of the first test case in the example:

T=0

				
				
				
				END

T=4

		 		
				
				
				END