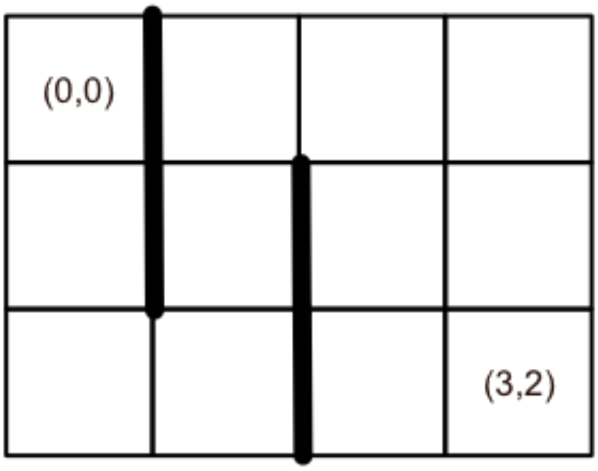
Jail

So you find yourself trapped in a prison. However, together with your inmates, you have deviced a plan to escape the prison guards and win your freedom!

In order to safely escape to the pickup car waiting, which will bring you and your comrades to freedom, you must cross the prison yard. The prison yard is a rectangular grid of dimensions *M* times *N* divided into unit squares. You and your inmate friends are at the *(0,0)* square, and you want to get to the pickup car which will be at *(M-1, N-1)*. However, the prison guards will be chasing you. When escaping, you will not necessarily take optimal decisions. In fact, you know you will always increase exactly one your coordinates by *1* at each step, running towards the rendez-vous point. This makes things a bit difficult, but luckily, your inmates will sacrifice themselves for your freedom. Whenever you encounter a wall, on of your inmates will stay behind to help all the others climb over.



Now, given a map of the prison yard you wish to calculate how many of you can make it.

Input:

The first line of input contains 3 integers, *M*, *N* and *I*, denoting dimensions of the prison yard, and the number of inmates escaping (including you). The following *N* lines contains *M-1* numbers that are all 0 or 1. There is a prison wall between square (i,j) and (i+1,j) if and only if the i'th number on the j'th line is 1. Then follows *N-1* lines with *M* numbers on each line, telling that there is a prison wall between (i,j) and (i, j+1) if and only if the i'th number on the j'th line is 1. Square coordinates and lines are here 0-indexed.

Output:

Output one number on a single line, the maximal number of inmates that may reach the pickup point (including you).

Constraints

$2 \le M, N \le 300$

$1 \le I \le 10^9$

Sample input 1

```
4 3 4
1 0 0
1 1 0
0 1 0
0 0 0 0
0 0 0 0
```

Sample output 1

```
3
```

Sample input 2

```
4 5 1
1 1 0
1 1 1
0 0 1
1 0 0
0 0 0
0 0 0 0
0 0 0 0
1 0 1 1
1 0 1 1
```

Sample output 2

```
1
```