Problem Description

In a given grid, each cell can have one of three values:

- Value 0 representing an empty cell
- Value 1 representing a fresh orange
- Value 2 representing a rotten orange

Every minute, any fresh orange that is adjacent (4-directionally) to a rotten orange becomes rotten.
Print the minimum number of minutes that must elapse until no cell has a fresh orange. If this is impossible, print -1 instead.

Input format

First line contains an integer T, representing number of test cases
For each test case, we'll have the following:
First line contains two space separated integers, M and N, where M and N represent the number of Rows and Columns in the grid respectively.
Next M lines contain N space separated integers consisting of 0, 1 or 2.

Output format

Print the minimum number of minutes that must elapse until no cell has a fresh orange. If this is impossible, print -1 instead.

Constraints

1 <= T <= 1000
0 <= N <= 1000
0 <= M <= 1000
0 <= Grid[i][j] <= 2
It's guaranteed that the sum of the number of elements across all test cases will be less than 500000.

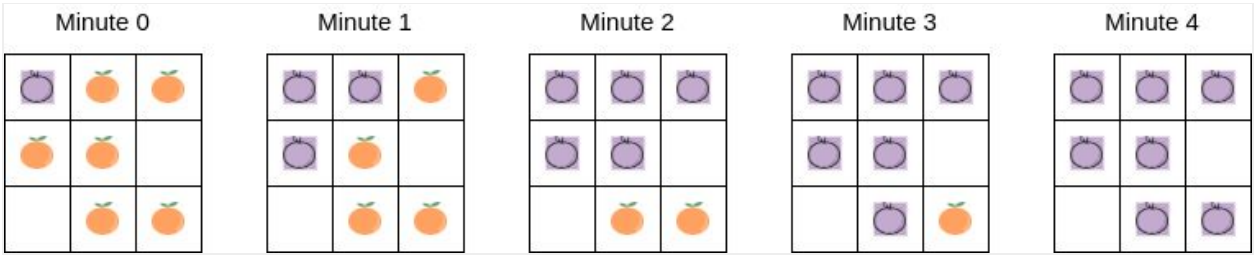
Sample Input 1

```
1
3 3
2 1 1
1 1 0
0 1 1
```

Sample Output 1

```
4
```

Explanation 1



Start with one rotten orange at (0,0). After 1 minute, the oranges at (0,1) and (1,0) will become rotten. After 2 minutes, the oranges at (0,2) and (1,1) will become rotten. After 3 minutes, the orange at (2,1) will become rotten. After 4 minutes, the last remaining fresh orange at (2,2) will become rotten. Overall, it takes 4 minutes for all fresh oranges to become rotten.

Sample Input 2

1
3 3
2 1 1
0 1 1
1 0 1

Sample Output 2

-1

Explanation 2

The orange in the bottom left corner (2,0) will never become rotten since it doesn't have any neighboring oranges in all 4 directions.

Sample Input 3

1
1 2
0 2

Sample Output 3

0

Explanation 3

Since there are no fresh oranges at minute 0 and the existing one is already rotten, the answer is 0.