# Problem C - Heap Like Sort

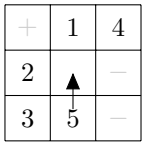
Little Heap likes to sort things. He likes to sort some sort of things using heaps. One day, Little Heap gets a large square board. Moreover, there are several distinct numbers on the anti-diagonal cells.

An **open cell** is an empty cell that there are no filled cell to its left or above or any north-west direction. A **closed cell** is an empty cell that there are no filled cell to its right or below or any south-east direction. An **inner cell** is any cell on the board that is neither an open cell nor a closed cell. For example, open cells below is marked by $+$, and closed cells are marked by $-$. It is possible for a cell that is recognized as open and closed at the same time. We call a cell a **open-only cell** if it is an open cell but not a closed cell.

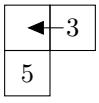
| | | | | | |
|---|---|---|---|---|---|
| + | + | + | + | + | 4 |
| + | + | + | + | 1 | − |
| + | + | + | 3 | − | − |
| + | + | 5 | − | − | − |
| + | 6 | − | − | − | − |
| 2 | − | − | − | − | − |

Little Heap start moving these numbers by the following algorithm:

1. Whenever there is an **inner empty cell** with its right cell and bottom cell filled with some integers (or a closed cell). Little Heap pick the smaller one among the two (or the only one) and drag them into this cell.




2. If there is no such empty cell described above, pick any one **open-only cell** that has *no* other open-only cells to its right or below. In this case there must be at least one cell to its right or below is filled. If after dragging the number from that filled cell, all numbers in the same row or same column remains increasing (from left to right, or from up to down), Little Heap can drag the number from the filled cell into it.




3. If there is no empty cell described in 1. or 2., the whole process stops.

Little Heap finds that if originally all the numbers are on anti-diagonal cells, then after continuously applying the first two rules, the numbers in the same row or the same column remain increasing (from left to right, or from up to down).

It is known that, no matter how you choose these empty cells in order, the outcome will always be the same.

Now Little Heap had performed some steps according to the algorithm, please continue his algorithm and output the number of non-empty cells in each rows after it stops.

## Input

The first line contains an integer $T$ ($1 \leq T \leq 100$), indicating the number of test cases.

For each test case, the first line contains two integers $n, m$ ($1 \leq n, m \leq 100$) indicating that we are showing the first $n$ rows and the first $m$ columns of the board.

Each of the next $n$ lines have $m$ integers separated by whitespaces. The $j$-th number in the $i$-th row $a_{ij}$ ($0 \leq a_{ij} \leq nm$) denote the number inside that cell. If $a_{ij} = 0$, it means that the cell is empty. All positive values will be distinct, and the input will be valid.

## Output

For each test case, output $K$ integers, one at a line: the number of non-empty cells in each non-empty row after the algorithm stops.

## Sample Input

```
3
3 3
0 0 1
0 2 0
3 0 0
6 6
0 0 0 0 0 4
0 0 0 0 1 0
0 0 0 3 0 0
0 0 5 0 0 0
0 6 0 0 0 0
2 0 0 0 0 0
3 5
```

```
1 2 3 4 5
6 7 0 0 0
8 0 0 0 0
```

## Sample Output

```
1
1
1
3
1
1
1
5
2
1
```