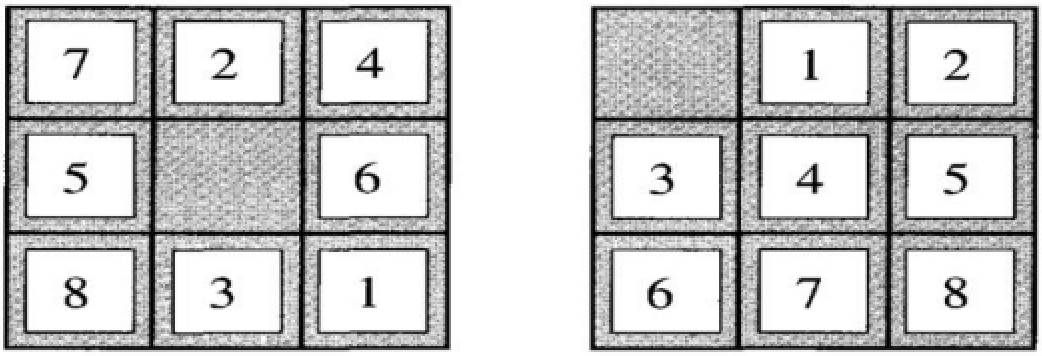
We know that the 8-puzzle is a sliding puzzle where there are numbered tiles in no particular order and only one square is empty. In each move, we can slide one adjacent tile to the empty space. Thus we need to place all the tiles in order. An example of initial state and goal state is shown here.



Start State

Goal State

Here, we are considering an easier version of the puzzle. In each move, any tile can be moved to the empty space, i.e. the adjacency is not required. In other words, we are allowed to swap the position of any tile with the empty space in one move. The goal is to calculate the minimum number of moves to solve the puzzle for a $N \times N$ board.

Input

The first line of input contains a positive integer T ($T < 100$). Then T cases follow, where the first line of each case contains a positive integer N ($2 \leq N \leq 500$). Then each of the following N lines contains N space separated integers denoting a row of the puzzle. The empty square is denoted by '`0`'. Each integer from 0 to $N^2 - 1$ (inclusive) will be present exactly once.

Output

For each case, print '`Case` X`:` Y' in a separate line, where X is the case number and Y is the minimum number of moves to solve the puzzle.

Sample Input

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

Sample Output

```
Case 1: 8
Case 2: 3
```