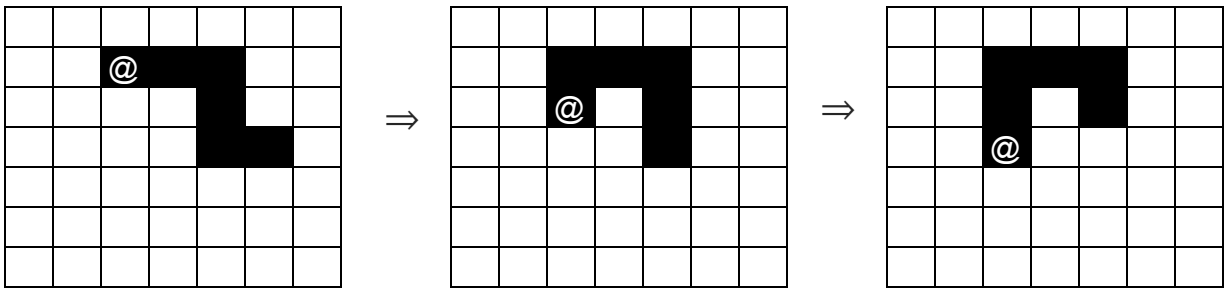
Snake Game (prob8)

The Problem

Snake is a video game in which a player controls a sequence of squares (the snake) on a limited 2 dimensional grid (field). Each square in a sequence is 4-connected to the previous one (they share a side on the grid). On each time step (e.g. each second) the first square of the sequence (the head of the snake) moves to one of 4 possible directions (left/right/up/down) and the rest of the snake follows it, i.e. the second square moves to the previous position of the head, the next square takes the old position of the second square and so on. The previous position of the last square of the snake becomes free.



Snake cannot go beyond the boundaries of the field, cannot go to cells with obstacles or over its own body, i.e. the head of the snake cannot move to a square that will be occupied by another part of its body after the move is complete. Note, that if the snake is longer than 3 squares it is possible for the head to go to a cell where the tail currently is, because the tail will also move and make its current position available.
In this problem the goal is to get to the specified “magical” cell. Of course playing games for you as a programmer is boring. It is much more fun to code. Thus let’s write a program to find the minimal number of steps the snake needs to reach the magical square.

Input

The input starts with a single integer T ($1 \leq T \leq 30$) - the number of test cases.

The first line of each test case contains 3 integers N, M, K. Where N and M are dimensions (number of rows and columns) of the field ($1 \leq N,M \leq 10$) and K is the length of the snake ($1 \leq K \leq 10$).

The next N lines describes the field and have M characters each. Each character can be:

'.' - a cell is free

'x' - a cell is a magical one. There will be exactly one magical square on the board.

'#' - a cell contains an obstacle and the snake cannot move to it

'0', .., '9' - parts of the snake. '0' is the head of the snake and the other parts are numbered sequentially. It is guaranteed, that the input is correct and the snake is "connected", i.e. head of the snake '0' is 4-connected to '1', which is 4-connected to '2', etc.

Output

For each i-th test case output "Game #i: <res>" (without quotes), where i is the 1-based number of test case and <res> is the minimal number of steps to reach the magical square or -1 if it is impossible (see sample output).

Sample Input

```
5
3 2 2
0.
1.
.x
3 3 2
.x.
##.
10.
5 5 5
....x
..432
....1
....0
.....
2 3 4
x10
.23
1 3 2
x10
```

Sample Output

```
Game #1: 3
Game #2: 4
Game #3: 5
Game #4: 4
Game #5: -1
```