Royal Access

Consider a *nxn* chessboard in which a king is placed in the centre of the board. However, the king is defective in the sense that it has lost the ability to move diagonally. That is, the king can only move to one adjacent cell per move, either: up, down, left, or right.

Your task is to determine which cells can be reached by the king in a given number of moves *m*.

There are two complications. Firstly, some of the cells contain an obstacle and cannot be moved into. Secondly, zero or two cells on the chessboard contain a teleporter. A king on a teleporter may move to the location of the other teleporter as one move. Of course, the king can also perform a normal (non-diagonal) move away from the teleporter.

The Input

The input will consist of up to 60 different test cases. For each test case, the first line contains the two integers *n* (where *n* is odd and 1 <= *n* < 50) and *m* (*m* < *n*) separated by one space. *n* lines follow, each containing *n* characters that represent one row of the board. Characters in a row can be either a '.' representing an empty cell, a '#' representing an obstacle, or a '@' representing a teleporter. You are guaranteed that the centre cell will always be free. The input is terminated with a test case where *n* and *m* are both 0.

The Output

For each test case, output *n* rows each containing the *n* characters that represent the reachable cells in *m* moves. Output '0' for a cell that cannot be reached and '1' for a cell that can be reached.

Differentiate the output for the different test cases by adding blank lines between them.

Sample Input

```
3 1
...
#..
...
5 3
....@
#####
.@...
.....
.....
0 0
```

Sample Output

```
010
011
010

00011
00000
11111
11111
01110
```