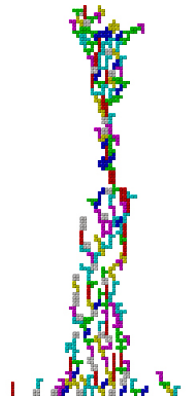
A Anti-Tetris

Time limit: 1s

In the game *Tetris*, the goal is to position blocks falling down a grid as well as possible. *Before* the block falls down, the player can shift the block to the left and right, and rotate it in steps of 90 degrees. Then, the block falls down vertically until it hits another block. Completely filling a row removes this row from the grid, clearing up space for more falling blocks.



The perfect example of a non-perfect *Tetris* grid. CC BY-NC 2.0 by Richard Buitron on Flickr

You have played this game one too many times, and to shake things up, you decide to play *Anti-Tetris*: instead of controlling the positioning of the blocks falling down, the goal is to design a Tetris grid that will perfectly fit a given block. That is, a grid such that after optimally positioning the new block, all rows of the grid are cleared and no filled cells remain.

As an example, consider the first sample case, shown in Figure A.1. The input block can be rotated clockwise 90 degrees and shifted left to make it fit exactly and clear all rows of the grid once it touches down.

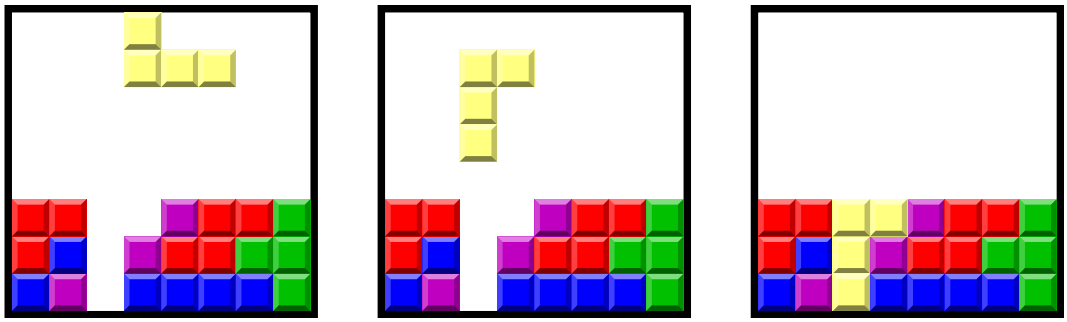


Figure A.1: Visualization of the first sample case. The falling block (the input, light yellow) perfectly fits in the Tetris grid (the output, other colours).

Input

The input consists of:

- One line with two integers h and w ($1 \leq h, w \leq 100$), the height and width of the Tetris block that is about to enter the grid.
- h lines with w characters, each character being either '`#`' or '`.`', representing a filled or unfilled cell of the block, respectively.

The input block is fully orthogonally connected and exactly fits in the $w \times h$ bounding box, i.e. the first and last row and column contain at least one '`#`'.

Output

If there exists no Tetris grid that perfectly fits the input block, output "`impossible`". Else, output a grid such that placing the the input block optimally removes all rows, in the following format:

- Two integers h, w ($1 \leq h, w \leq 1000$), the height and width of the Tetris grid.
- h strings with w characters, each character being either '`#`' or '`.`', representing a filled or unfilled cell in the Tetris grid, respectively.

A row in the output grid may not be completely filled before the block is added, since such a row would already have been removed by the game.

Note that it is not required to print empty rows at the top of the output grid, since the block can be rotated and shifted to the left and right *before* it falls down.

If there are multiple valid solutions, you may output any one of them.

Sample Input 1

```
2 3
#..
###
```

Sample Output 1

```
3 8
##..####
##.#####
##.#####
```

Sample Input 2

```
2 3
.##
##.
```

Sample Output 2

```
impossible
```

Sample Input 3

```
3 3
#..
##.
###
```

Sample Output 3

```
5 7
.......
.......
##...##
###..##
####.##
```