Problem E
Even Paths

Last year, Arthur participated in the Vietnamese Robotic Olympiad with a malfunctioning robot. This year, he decided to participate one more time.

The challenge for contestants this year is much harder. The playing field is a rectangular board of size $r \cdot c$, with r rows numbered from 1 to r from top to bottom and c columns numbered from 1 to c from left to right. The cell which lies on ith row and jth column is denoted as (i, j). Each cell contains either a light or a stone, but not both. These lights can be turned on or off. Initially, some of them are on.

The challenge's objective is to switch off all the lights. In order to do so, contestants' robots can execute several runs. In each run, a robot starts at an arbitrary cell on the board, and then performs a sequence of moves. There are several rules for a valid run:

- In each step, a robot has to move from the current position to a side-adjacent cell. In other words, it has to move to the next cell towards any of the four directions: up, down, left or right.
- Robots must always stay inside the board.
- Robots must not visit any cells containing stones.
- In a run, a robot can not visit a cell more than once. However, a robot can visit a cell multiple times during different runs.
- The number of visited cells in a run **must be even**.

From all the above rules, it can be easily seen that a valid run is uniquely defined by a tuple (x, y, s), where (x, y) represents its starting position, and s is a string containing characters **U**, **D**, **L**, **R** representing its sequence of moves. Here **U**, **D**, **L**, **R** stands for up, down, left and right, respectively. The length of the string s **must be odd**.

According to the rules of the Olympiad, while executing a run, whenever a robot passes through a cell, it can choose to turn off the light there. However, as Arthur's robot is buggy again, his robot decides to **always switch the light** (on to off, off to on) when it visits a cell. In other words, all the lights of all cells on its path will be switched.

With this buggy robot, it is even hard just to turn off all the lights. Your task is to help Arthur to accomplish that. He will be extremely thankful if you manage to turn off all the lights. The number of runs does not need minimizing, but the number of moving steps over all runs should not exceed 10^6.

Input

The input starts with an integer t — the number of test cases ($t \leq 100\,000$). Then t test cases follow, each test case is presented as below:

- The first line contains two integers r and c ($1 \leq r, c \leq 10^6$) representing the size of the board.

- The last r lines represent the board. Each contains a string of length c, whose characters are either `.`, `X` or `#`, where:
  - character `.` represents a cell with a light initally turned off,
  - character `X` represents a cell with a light initally turned on,
  - character `#` represents a cell with a stone.

The sum of $r \cdot c$ over all test cases of an input does not exceed 10^6.

Output

For each test case:

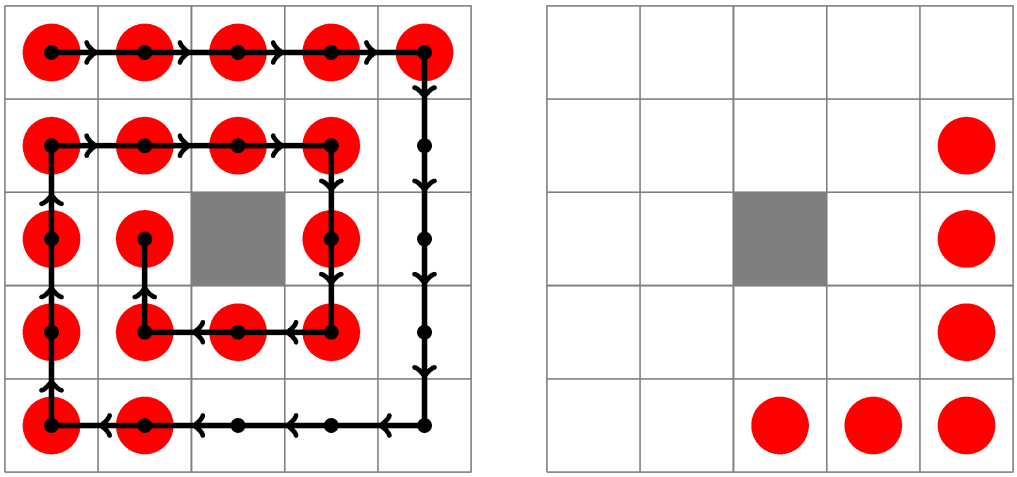
- If it is impossible to turn off all the lights using at most 10^6 moving steps, print -1.
- Otherwise, the first line contains an integer k $(0 \le k \le 10^6)$ representing the number of runs. Then k lines follow, each contains two integers x, y and a string s to describe a run as presented above. The length of all these strings should not exceed 10^6.

If there are multiple solutions, you can output any of them.

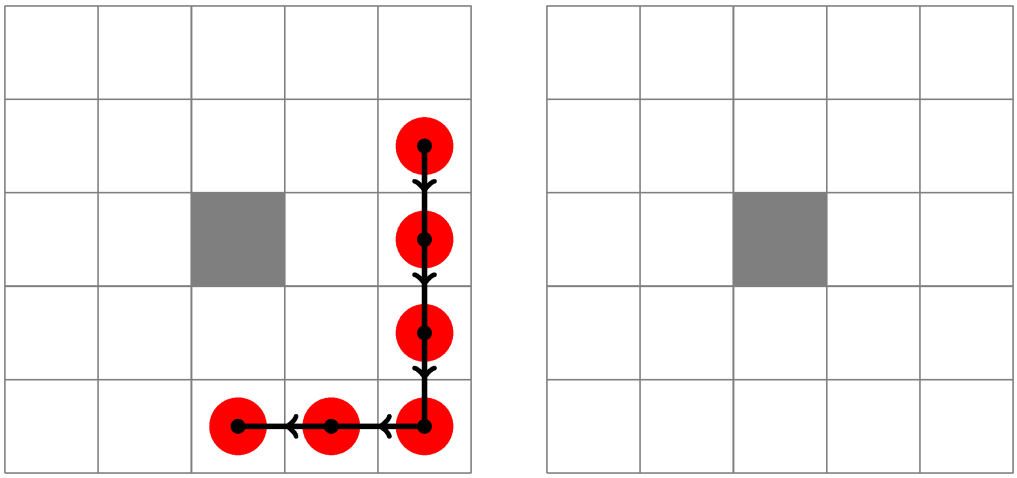
Explanation of the sample

The following figures demomstrate the above sample. In each figure, red circles represent lights which are turned on, blank cells represent cells with lights which are turned off, and grey cells represents cells with stones. Black bold segments and arrows represent runs.

These two figures below demonstrate the first run. The left one represents the state of the board before the run and how robot moves during the run, while the right one represents the state of the board after the run.



These two figures below demonstrate the second run. The left one represents the state of the board before the run and how robot moves during the run, while the right one represents the state of the board after the run.



Sample Input 1	Sample Output 1
1 5 5 XXXXX XXXX. XX#X. XXXX. XX...	2 1 1 RRRRDDDDLLLLUUURRRDDLLU 2 5 DDDLL