时间限制：C/C++/Rust/Pascal 2秒，其他语言4秒
空间限制：C/C++/Rust/Pascal 1024 M，其他语言2048 M
Special Judge, 64bit IO Format: %lld

题目描述

Sokoban is a puzzle video game franchise created by Hiroyuki Imabayashi in 1981, in which players push boxes around a warehouse to designated storage locations. Despite its simple rules, Sokoban exhibits intricate and challenging behavior that has captivated both puzzle enthusiasts and theoretical computer scientists. In 1996, Dorit Dor and Uri Zwick proved that Sokoban is NP-complete, and in 1997, Joseph C. Culberson further showed that it is PSPACE-complete. To this day, Sokoban remains popular among players, and many modern puzzle games are inspired by its mechanics, including Stephen's Sausage Roll, Baba Is You, and Patrick's Parabox.

Bobo was also recently playing Sokoban and found it incredibly hard to complete a level. Feeling frustrated, he complained, "Mature boxes should learn to push themselves!" And then, bing! The magic happened. Suddenly, the pusher disappeared from Sokoban, and you can order boxes to push themselves. Now, the game becomes much easier! ... Or does it?

Formally, you need to solve the following problem of Sokoban with No Pushers:

You are given an $n \times m$ two-dimensional grid surrounded by walls, which may also contain additional internal walls. There are several boxes and target positions on the grid. **The number of boxes equals the number of target positions**. It is guaranteed that no box or target position occupies a wall cell, and that all boxes and all target positions are located at distinct places.

You may repeatedly perform the following operation:

- Select any box and move it one square in one of the four cardinal directions (up, down, left, or right), provided the destination is not a wall or another box.

Your task is to determine whether it is possible to move all boxes onto target positions. If so, construct one such sequence of moves.

输入描述:

The first line of input contains two integers n, m $(1 \leq n, m \leq 50)$, denoting the height of the map, the width of the map, and the number of boxes, respectively.

Then n lines follow, where each line contains a string of m characters, denoting the map. Each character lies within the set of {`#', `.', `@', `*', `!'}, such that for the j-th $(1 \leq j \leq m)$ character of the i-th $(1 \leq i \leq n)$ string:

- a `#' character denotes that a **wall** is at position (i, j);
- a `.' character denotes that position (i, j) is **vacant**;

C++（clang++18）
1
ACM模
请通过
入输出
出描述
运行结果 自测