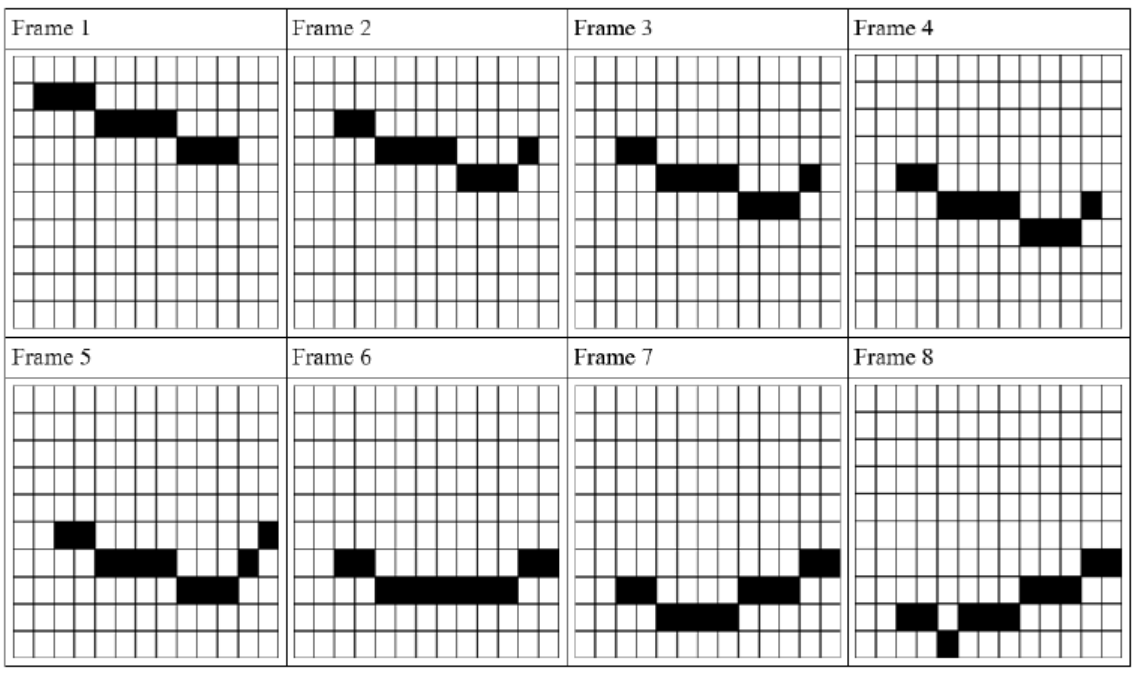
Most animation tools that support 2D animation support defining some keyframes where the user will specify what the image will like at a particular time for that particular frame. The user tells the tool how many frames to use (or how long the animation will run). The tool would then generate the intermediate frames so that the transition from one frame to another looks smooth.

For this particular problem, we would address the problem from a different perspective. We would supply a source frame and a target frame as keyframes. The problem is to find out the minimum number of frames needed to go from source to target.

To keep it simple, we are only dealing with bitmap keyframes. Here each pixel will be either black or white. The pixels can travel from one location to another from frame to frame — infact, multiple black pixels can travel going from one frame to the next one. Each pixel can move one cell up, down, left or right. While traveling a pixel is allowed to collide with another. As the target keyframe can have more (or less) black pixels compared to the source keyframe, we need some sort of mechanism to generate new (or to eat up existing) black pixels. We will allow a black pixel in one frame to turn any combination of it's four neighboring pixels (up, down, left and right) black to generate new black pixels when needed. We will also allow a black pixel in one frame to turn itself white in the next frame if needed. Without the help of a neighboring black pixel, a pixel won't turn black by itself. To further simplify, we'll also assume that the source and target key frames will have at least one black pixel. For example, a black pixel can turn its top, left and right pixels black (not modifying the bottom pixel) and turn itself white — all of this going from one frame to the next. However, a black pixel cannot travel two pixels to the right in one frame as it's allowed to travel only one cell in a frame.

The following illustration may help explain the process:



Input

There can be several test cases (less than 500). Each test case will start with 2 integers, the number of rows, R ($1 \leq R \leq 100$) and columns C ($1 \leq C \leq 100$) for the source and target bitmap keyframes. The next R lines will contain C characters each, defining the source keyframe. The R lines right after that will define the target keyframe in the same fashion. Each character of the keyframe will either be a '`.`' denoting white pixels or be '`#`' denoting a black pixel. We can assume that the maximum number of black pixels in a keyframe will be less than 4,000. A test case with $R = 0$ and $C = 0$ will signify the end of input.

Output

Output for each test case will start with the case label in the format '`Case` k`:  `' where k is the case number (starting from 1) followed by the number of frames needed to go from the source keyframe to the target keyframe.

Sample Input

```
10 13
.............
.............
.###.........
....####.....
........###..
.............
.............
.............
.............
.............
.............
.............
.............
.............
.............
...........##
........###..
.....###.....
..###........
2 2
..
.#
..
.#
0 0
```

Sample Output

```
Case 1: 7
Case 2: 0
```