Problem B. Always Turn Left

Small input 40 points	Solve B-small
Large input 80 points	Solve B-large

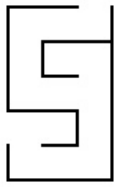
Problem

You find yourself standing outside of a perfect maze. A maze is defined as "perfect" if it meets the following conditions:

1. It is a rectangular grid of rooms, **R** rows by **C** columns.
2. There are exactly two openings on the outside of the maze: the entrance and the exit. The entrance is always on the north wall, while the exit could be on any wall.
3. There is exactly one path between any two rooms in the maze (that is, exactly one path that does not involve backtracking).

You decide to solve the perfect maze using the "always turn left" algorithm, which states that you take the leftmost fork at every opportunity. If you hit a dead end, you turn right twice (180 degrees clockwise) and continue. (If you were to stick out your left arm and touch the wall while following this algorithm, you'd solve the maze without ever breaking contact with the wall.) Once you finish the maze, you decide to go the extra step and solve it again (still always turning left), but starting at the exit and finishing at the entrance.

The path you take through the maze can be described with three characters: 'W' means to walk forward into the next room, 'L' means to turn left (or counterclockwise) 90 degrees, and 'R' means to turn right (or clockwise) 90 degrees. You begin outside the maze, immediately adjacent to the entrance, facing the maze. You finish when you have stepped outside the maze through the exit. For example, if the entrance is on the north and the exit is on the west, your path through the following maze would be `WRWWLWWLWWLWLWRRWRWWWRWWRWLW`:



If the entrance and exit were reversed such that you began outside the west wall and finished out the north wall, your path would be `WWRRWLWLWWLWWLWWRWWRWWLW`. Given your two paths through the maze (entrance to exit and exit to entrance), your code should return a description of the maze.

Input

The first line of input gives the number of cases, **N**. **N** test cases follow. Each case is a line formatted as

```
entrance_to_exit exit_to_entrance
```

All paths will be at least two characters long, consist only of the characters 'W', 'L', and 'R', and begin and end with 'W'.

Output

For each test case, output one line containing "Case #**x**:" by itself. The next **R** lines give a description of the **R** by **C** maze. There should be **C** characters in each line, representing which directions it is possible to walk from that room. Refer to the following legend:

Character	Can walk north?	Can walk south?	Can walk west?	Can walk east?
1	Yes	No	No	No
2	No	Yes	No	No
3	Yes	Yes	No	No
4	No	No	Yes	No
5	Yes	No	Yes	No
6	No	Yes	Yes	No
7	Yes	Yes	Yes	No

8	No	No	No	Yes
9	Yes	No	No	Yes
a	No	Yes	No	Yes
b	Yes	Yes	No	Yes
c	No	No	Yes	Yes
d	Yes	No	Yes	Yes
e	No	Yes	Yes	Yes
f	Yes	Yes	Yes	Yes

Limits

1 ≤ **N** ≤ 100.

Small dataset

2 ≤ len(entrance_to_exit) ≤ 100,
2 ≤ len(exit_to_entrance) ≤ 100.

Large dataset

2 ≤ len(entrance_to_exit) ≤ 10000,
2 ≤ len(exit_to_entrance) ≤ 10000.

Sample

```
Input
2
WRWWLWWLWWLWLWRRWRWWWRWWRWLW WWRRWLWLWWLWWLWWRWWRWWLW
WW WW

Output
Case #1:
ac5
386
9c7
e43
9c5
Case #2:
3
```