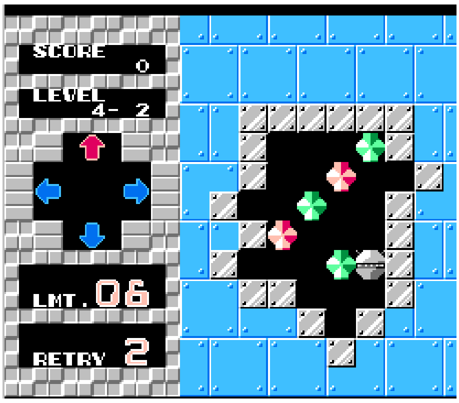

Tomy's fond of a game called 'Fans and Gems' (also known as **Gravnic**). In the game, he can use fans to collect gems, but he's satisfied with his play only if all the gems are collected with minimal number of steps. The game is played as following:

There are three kinds of gems, one colored red, one colored green and one colored blue. There are walls in the space, as you see. There are also virtual fans everywhere in the game, but you cannot see them. What you can do each time is to select a DIRECTION to which the fans should blow. There are only four directions possible: UP, DOWN, LEFT and RIGHT. Then, the fans will work, pushing all the gems to fly to the selected direction at the same speed, until they cannot move further(blocked by the wall, other gems or a flyer). Then, if there are some gems touching some same-colored gem(touching means adjacent in one of the four directions), they disappear simultaneously. Note that the fans are still working, so some gems may go on moving in that direction after some gems disappeared. The fans stop working when all the gems cannot move any more, and none of them should disappear. There may be some flyers that can be moved by the fans, but they can NEVER disappear.

You are to write a program that finds the minimal number of operations to make all the gems disappear.

Input

The input file begins with an integer T, indicating the number of test cases ($1 \leq T \leq 50$). Each test case begins with two integers N, M, indicating the height and width of the map ($1 \leq N \leq 12$, $1 \leq M \leq 20$). In the following N lines, each line contains M characters describing the map. There is one line after each map, ignore them. Spaces denotes empty square, '`#`' denotes a wall, '`1`' denotes a red gem, '`2`' denotes a green gem, '`3`' denotes a blue gem, and '`@`' denotes a flyer. It's guaranteed that the four sides of the map are all walls. There is at least one gem in the map, and no two same-colored gems will touch each other at the beginning of the game.

Output

You should print a single line for each case. If there is a solution, write the shortest operation sequence in a string. The ith character must be from the set {'`U`','`D`','`L`','`R`'}, describing i-th operation. The four characters represent UP, DOWM, LEFT, RIGHT respectively. If there are more than one solution, choose the lexicographical smallest one, if there are no solution, output '`-1`' on the line. When a solution exists, you need only 18 or fewer steps to carry it out.

Sample Input

```
2
9 8
########
##   1##
##  2  #
#  1  ##
##2   ##
#   1@##
###   ##
#### ###
########

7 8
########
#212121#
#      #
# # # ##
# #    #
#     ##
########
```

Sample Output

```
LURD
DL
```