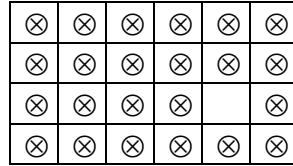


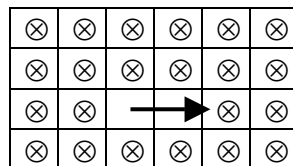
Problem 4: Coke Bottles

A long time ago - before aluminum cans and plastic bottles, Coke used to actually come in glass bottles. A case of Coke consisted of a box or carton that had 24 bottles, arranged as 4 rows of 6 bottles each. Unfortunately in this box one bottle is missing:

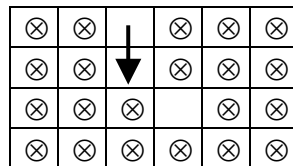


In this instance the missing bottle comes from row 3 column 5; the rows are numbered from 1 to 4, and the columns are numbered from 1 to 6.

We will play a game based on this diagram. You can "jump over" a bottle, either horizontally or vertically, and remove the bottle you have jumped over. That is, you can move a bottle over another bottle to an empty space in the box, and then remove the bottle over which you jumped. For example, I can take the bottle in row 3 column 3, jump over the one in row 3 column 4, and land in row 3, column 5. Since I can do this, I can remove the bottle from row 3 column 4:



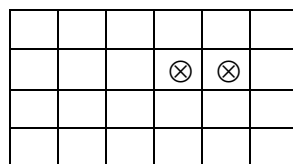
Now the next move from here might be to jump from row 1 column 3, over row 2 column 3, and into row 3 column 3, removing the bottle in row 2 column 3:



Possible moves from here include:

- [3,2] to [3,4] removing [3,3] (moving right from [3,2])
- [4,3] to [2,3] removing [3,3] (moving up from [4,3])
- [3,6] to [3,4] removing [3,5] (moving left from [3,6])
- etc.

Can you eventually get to a point where there is only one bottle left? The very last move might be something like this, where we can move from row 2 column 5 over row 2 column 4, and into row 2 column 3 (left from [2,5]):



In this problem you are given the description of a hypothetical Coke case containing fewer than 10 rows and 10 columns, the position (row and column) of a missing bottle, and the maximum number of bottles that can be left. Your job is to determine a sequence of "jumps" that will achieve this task. The valid moves are to jump up and over, down and over, left and over, or right and over. Diagonal jumps are not allowed; you may not drink the Cokes to create empty spaces.

Input

The input will contain multiple cases. For each case the input will contain five integers. The first two integers specify the number of rows and columns of the Coke case; neither of these will be larger than 9 or smaller than 1. The second two integers specify the row and column where a bottle is initially missing. The last integer specifies the maximum number of bottles that may be left. The last input case will be followed by five zeroes.

Output

For each case, first display the case number (they start with 1 and increase sequentially). If there is no solution for the case, then display the message "No solution". Otherwise, starting on the next line, display the sequence of "jumps" that will leave just the specified number of bottles in the case. Each jump is specified in the form (r,c,dir), where r and c specify the row and column number of the bottle that jumps (NOT the jumped bottle), and dir is the direction that the bottle moved (U, D, L, or R for up, down, left, or right). Display these in the style shown in the sample output below, with as many jumps per line as possible, but with no more than 80 characters per output line, including four blank characters of indentation at the beginning of each line. Separate the output for consecutive cases with a blank line. If there are multiple valid jump sequences, any of them is acceptable.

Sample Input

```
4 6 1 1 1
1 3 1 1 1
3 3 2 2 1
3 3 1 1 4
0 0 0 0 0
```

Output for the Sample Input

```
Case 1:
    (3,1,U) , (3,3,L) , (4,1,U) , (4,3,L) , (4,5,L) , (3,5,L) , (1,5,D) , (3,6,L) ,
    (3,4,L) , (2,3,R) , (1,6,D) , (4,6,U) , (2,6,L) , (1,4,D) , (1,1,D) , (4,1,U) ,
    (1,3,L) , (1,1,D) , (3,1,R) , (4,3,U) , (2,2,R) , (3,4,U)

Case 2:
    (1,3,L)

Case 3: No solution

Case 4:
    (3,1,U) , (3,3,L) , (1,3,D) , (1,2,D)
```