Hey, Have You Seen My Kangaroo?

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	128 megabytes

Please note the UNUSUAL MEMORY LIMIT of this problem.

After the great success in 2018, 2019, 2020, 2021, 2022, and 2023, the Nanjing University of Aeronautics and Astronautics (NUAA) will host the *International Collegiate Programming Contest* (ICPC) Nanjing regional for the seventh time in a row.

Team ***Power of Two*** and team ***Three Hold Two*** won the champion title for Tsinghua University in 2018 and 2019. In 2020, 2021, and 2022, team ***Inverted Cross*** from Peking University won the three-peat champion titles. In 2023, another team ***Reborn as a Vegetable Dog*** from Peking University won the title. They also won the 46th ICPC World Champion, reclaiming the trophy for the EC region after 13 years!

This year, around 335 teams are participating in the contest. At most 33 gold medals, 66 silver medals, and 99 bronze medals will be awarded (note that these numbers are for reference only). We are looking forward to seeing the participants' outstanding performance! We also want to express our gratitude for the hard work done by all staff and volunteers for this contest. Thank you all for your great contribution to this contest!

Photo taken in the 2023 ICPC Asia Nanjing Regional Contest

In the 2018 contest, problem K, ***Kangaroo Puzzle***, requires the contestants to construct an operation sequence for the game:

> The puzzle is a grid with n rows and m columns ($1 \le n, m \le 20$), and there are some (at least 2) kangaroos standing in the puzzle. The player's goal is to control them to get together. There are some walls in some cells, and the kangaroos cannot enter the cells with walls. The other cells are empty. The kangaroos can move from an empty cell to an adjacent empty cell in four directions: up, down, left, and right.
>
> There is exactly one kangaroo in every empty cell in the beginning, and the player can control the kangaroos by pressing the buttons U, D, L, R on the keyboard. The kangaroos will move simultaneously according to the button you press.
>
> The contestant needs to construct an operating sequence of at most 5×10^4 steps consisting of U, D, L, R only to achieve the goal.

In the 2020 contest, problem A, ***Ah, It's Yesterday Once More***, requires the contestants to construct an input map to hack the following code of the problem described before:

```
#include <bits/stdc++.h>
using namespace std;
string s = "UDLR";
int main()
{
  srand(time(NULL));
  for (int i = 1; i <= 50000; i++) putchar(s[rand() % 4]);
  return 0;
}
```

Furthermore, in the 2021 contest (Problem A, ***Oops, It's Yesterday Twice More***), the 2022 contest (Problem A, ***Stop, Yesterday Please No More***), and the 2023 contest (Problem A, ***Cool, It's Yesterday Four Times More***), every year we have a problem related to the kangaroos! We would like to introduce all these problems to you, but if we do so every year, we may have a 500-page statement for one single problem in the 3024 contest. Therefore, we omit them this time. Besides, you may already have seen them in the practice contest.

Now, in the 2024 contest, as everyone expects, the kangaroo problem is back again! We don't know why problem setters are so obsessed with kangaroos, but the problem is as follows:

You are given a grid with n rows and m columns. There are some walls in some cells, and the kangaroos cannot enter the cells with walls. The other cells are empty and each contains a kangaroo. The kangaroos can move from an empty cell to an adjacent empty cell in four directions: up, down, left, and right.

You can control the kangaroos by pressing the buttons U, D, L, R on the keyboard. The kangaroos will move simultaneously according to the button you press. Specifically, for any kangaroo located in the cell on the i-th row and the j-th column, indicated by (i, j):

1. Button U: it will move to $(i-1, j)$ if $i > 1$ and $(i-1, j)$ is not a wall. Otherwise, it will stay in the same cell.
2. Button D: it will move to $(i+1, j)$ if $i < n$ and $(i+1, j)$ is not a wall. Otherwise, it will stay in the same cell.
3. Button L: it will move to $(i, j-1)$ if $j > 1$ and $(i, j-1)$ is not a wall. Otherwise, it will stay in the same cell.
4. Button R: it will move to $(i, j+1)$ if $j < m$ and $(i, j+1)$ is not a wall. Otherwise, it will stay in the same cell.

You are given an operating sequence $s_1 s_2 \ldots s_k$ consisting only of characters 'U', 'D', 'L', and 'R'. The operations are performed infinitely according to the sequence. Specifically, if $1 \le t \le k$, the t-th operation is s_t; Otherwise if $t > k$, the t-th operation is the same as the $(t-k)$-th operation. For each $1 \le i \le n \times m$, find the smallest integer v_i such that after performing v_i operations, at most i cells will contain kangaroos.

Input

There is only one test case in each test file.

The first line contains three integers n, m, and k ($1 \le n, m \le 2 \times 10^5$, $1 \le n \times m \le 2 \times 10^5$, $1 \le k \le 200$), indicating the number of rows and columns of the grid, and the length of the operating sequence.

The second line contains a string $s_1 s_2 \cdots s_k$ ($s_i \in \{$'U', 'D', 'L', 'R'$\}$), indicating the operating sequence.

For the following n lines, the i-th line contains a binary string $a_{i,1} a_{i,2} \cdots a_{i,m}$ ($a_{i,j} \in \{$'0', '1'$\}$). If $a_{i,j} =$ '1' then cell (i, j) is empty; Otherwise if $a_{i,j} =$ '0' then cell (i, j) is blocked and cannot be entered. It is guaranteed that there is at least one empty cell in the grid.

Output

Output $n \times m$ lines, where the i-th line contains an integer v_i, indicating the minimum number of operations needed so that at most i cells will contain kangaroos. If this is impossible, just output `-1` on this line.

Examples

standard input	standard output
3 3 6 ULDDRR 010 111 010	-1 4 2 1 0 0 0 0 0
3 3 6 ULDDRR 010 111 011	7 4 2 1 1 0 0 0 0
1 5 1 R 11111	4 3 2 1 0