# Problem H. CHESSBOARD
# Time Limit: 5 seconds

## Problem Description

Ruirui and Doc are playing an interesting game on a chessboard with $n$ rows and $m$ columns. The rows are numbered from 1 to $n$ from top to bottom, and the columns are numbered from 1 to $m$ from left to right. There are some broken grids on the chessboards, which a chess cannot move in. Firstly, Doc gives Ruirui a sequence of commands, each command is of one of four following forms:

- **Move Up**: moving from grid $(x, y)$ to grid $(x - 1, y)$;
- **Move Down**: moving from grid $(x, y)$ to grid $(x + 1, y)$;
- **Move Left**: moving from grid $(x, y)$ to grid $(x, y - 1)$;
- **Move Right**: moving from grid $(x, y)$ to grid $(x, y + 1)$.

Then Ruirui puts a single chess on a grid of the chessboard

Ruirui will move the chess by Doc's commands in sequence. If the chess will be out of boarder or in a broken grid after a move, she omits this command and **go on** to consider the next one until the last command. Now Ruirui wants to find the grid which the chess will be in the end.

## Input

The first line contains a single integer $T$ ($1 \le T \le 10$), which indicates the number of test cases. Then $T$ test cases follow.

For each test case, the first line contains 4 integers $n, m, o$ and $l$ ($1 \le n, m, o, l \le 1000$) representing the number of rows, the number of columns, the number of broken grids and the length of Doc's command sequence.

Next $o$ lines, each line contains two integers $i$ and $j$ describing the position of broken grid.

The last line contains Doc's command sequence, it's a string of length $l$ with each character being one of {"$U$", "$D$", "$L$", "$R$"} denoting Move Up, Move Down, Move Left and Move Right respectively.

## Output

For each test case, for each unbroken grid $(i, j)$, assume a chess started at $(i, j)$ would stop at $(x(i, j), y(i, j))$, output the sum of $(i - x(i, j))^2 + (j - y(i, j))^2$ (over all unbroken $(i, j)$).

## Sample Input

```
2
5 5 5 5
2 3
5 1
5 5
4 4
3 5
RRRLR
10 10 10 10
2 6
3 8
7 2
5 3
4 3
3 2
7 9
6 8
9 10
10 6
DLLDRRURLR
```

## Sample Output

```
Case #1: 49
Case #2: 241
```