Assignment 3

Contributor List

> Leader: Qing WANG
>
> Problem 1: Sirui LI
>
> Problem 2: Yitong WANG
>
> Problem 3: Yitong WANG
>
> Problem 4: Yuwei XIAO

Problem 1. Mine Sweeper(Easy, 20 points)

Siri is a CSE student who loves playing games, recently he is obsessed with *Mine Sweeper*. Unfortunately, he is too dumb to understand the game rule, so he asks you to help him develop a program that can mimic the process of this game.

Before you develop this program, here are the preliminaries you should know:

- The game board provide to you will be in the form of $n \times n$, where mine squares will be presented in lower case English letter 'x' and safe squares will be presented in lower case English letter 'o'.
- You will be provided with a specific coordinate which indicates a square on the board. The output must be the detail of the corresponding square. If the square is a mine square, you should output '-1', otherwise you must calculate how many mines are surround it and output the number of surrounding mines.

By finishing this problem, we believe you will be an expert on *Mine Sweeper* one day!

Input Format

The first line will be an integer n, which indicates the side length of game board.

For the next n lines that represent the content of game board, each line will be a string of length n.

The last line will be the coordinate.

Output Format

A number indicates the detail of the specific square.

Sample with Explanation

Input

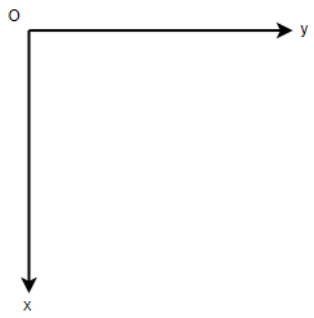
```
5
ooooo
ooxoo
oxooo
ooooo
ooooo
2 2
```

Output

```
2
```

Hint

1. This coordinate will be set base on the coordinate axis below:



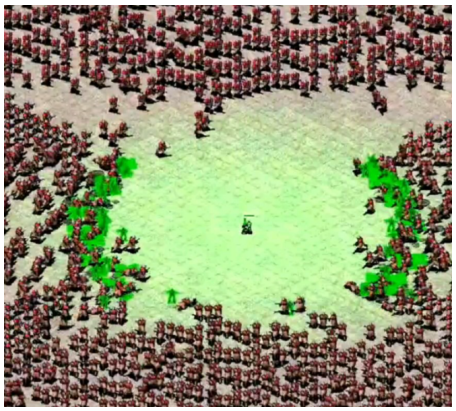
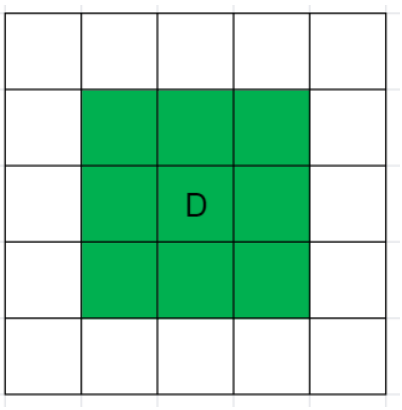
2. For 100% testcases, $1 \leq n \leq 100, 0 \leq x, y < n$.

Problem 2. Safe Areas in RA2 Battlefield(Medium, 20 points)

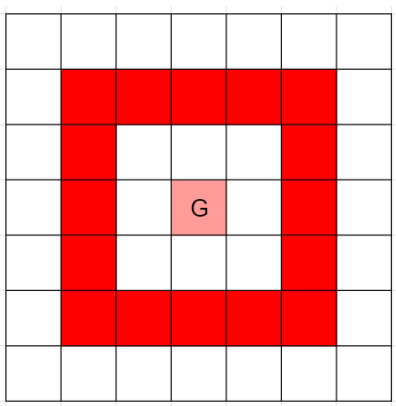
Recently, thanks to a series of video uploaders such as WM=HBK08(红警HBK08) and =HY=Moon3(红警月亮3), Red Alert 2 becomes a popular game in China. Suppose you are a commander and want to drop paratroopers(伞兵) in the $n \times n$ battlefield of Red Alert 2. However, there are m Iraqi Desolators(伊拉克辐射工兵) and k French Grand Cannons(法国巨炮) of enemies. Now you are given the positions of Desolators and Grand Cannons and need to find out the number of safe areas in the $n \times n$ battlefield to drop your paratroopers.

Here is the introduction to Desolators and Grand Cannons in this problem:

- Desolator: The Desolator uses a radiation cannon to irradiate the ground around him(blocks in green), damaging all units around him.

- Grand Cannon: The Grand Cannon is a larse artillery gun firing powerful heavy shells to an extreme range and causes considerable damage in the target area(blocks in red). However, it has a minimum range and cannot damage the enemy around it. Considering it is a defensive tower, the area captured by it(block in pink) is also unsafe.

Input Format

The first line gives n, m, k.

Then the following m lines define x_i, y_i, the positions of Desolator.

Then following k lines define o_i, p_i, the positions of Grand Cannons.

Output Format

The number of safe areas for droping paratroopers in the battlefield of Red Alert 2.

Sample with Explanation

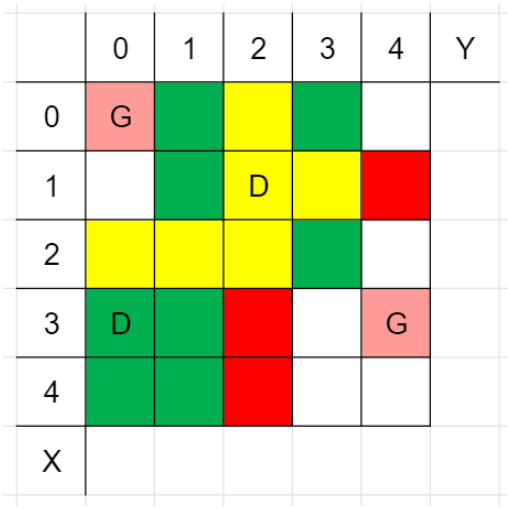
Input

```
5 2 2
3 0
1 2
0 0
3 4
```

Output

```
6
```

Explanation



The battlefield is shown in the figure. The blocks in green are irradiated by Desolators and those in red can be damaged by Grand Cannons. The blocks in yellow can be both damaged by Desolators and Grand Cannons.
Considering blocks in pink i.e. (0,0) and (3,4) are captured by Grand Cannons, there are 6 safe areas i.e. (0, 4), (1, 0), (2, 4), (3, 3), (4, 3) and (4, 4).

Hint

1. You can try to make use of `boolean[][]`.
2. It is recommended to do the boundary checking before accessing array element.
3. Test data scale:

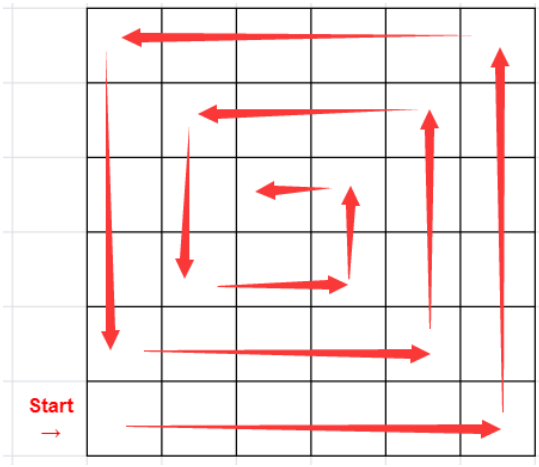
- For 10% testcases, $n = 1$;
- For another 20% testcases, $m = 0$ or $k = 0$;
- For 60% testcases, $0 \leq x_i, y_i, o_i, p_i < n$;
- For 100% testcases, $1 \leq n \leq 100, 0 \leq m, k \leq 25, 0 \leq m + k \leq 25$, $-50 \leq x_i, y_i, o_i, p_i \leq 150$.

Problem 3. Compressed Spiral Matrix(Hard, 30 points)

In this problem, you are required to fill in the matrix of size $m \times n$.
You are given matrix size m, n and a compressed String s containing lower letters and numbers.
And you need to do the following tasks in this problem:

(1) Decompress the compressed String. The String s is compressed in the form of letters c_i and repeat times n_i. After decompression, the length of String s' is guaranteed to equal $m * n$ in our testcases i.e. $\sum_{i=0} n_i = length(s') = m * n$. For instance, `a6` can be decompressed into `aaaaaa` since $c_0 = a$ and $n_0 = 6$, and `a1b4` can be decompressed into `abbbb` since $c_0 = a$ and $n_0 = 1$, $c_1 = b$ and $n_1 = 4$.

(2) Organize the spiral matrix counterclockwise using the decompressed String s', starting from the bottom left corner.



(3) Print out the alphabet matrix after decompression.

Input Format

The first line gives m and n, and the second line gives the compressed String s.

Output Format

The spiral matrix after decompression.

Sample with Explanation

Input

```
4 5
a1b2c3d4e5f4g1
```

Output

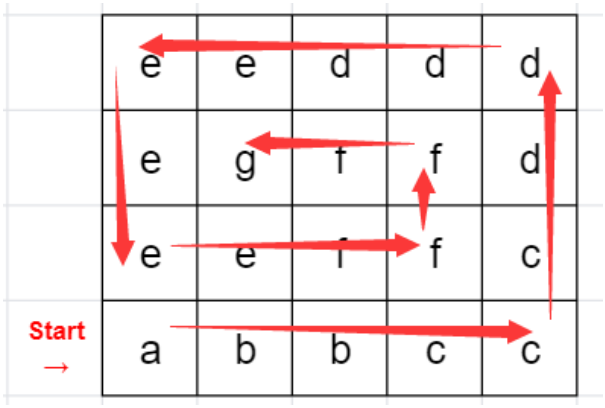
```
eeddd
egffd
eeffc
abbcc
```

Explanation

First, we do the decompression for the given String s.

```
Before: a1b2c3d4e5f4g1
After: abbcccddddeeeeeffffg
```

Next, we start from the bottom left corner and organize the 4×5 spiral matrix counterclockwise using s'(abbcccddddeeeeeffffg).



Finally, we print the matrix out and get the output in the sample.

```
eeddd
egffd
eeffc
abbcc
```

Hint

1. It is recommended to use methods to simplify this problem.
2. Test data scale:

- For 10% testcases, $n_i = 1$ for the compressed String s;
- For another 10% testcases, there is only c_0 and n_0 in the compressed String s;
- For 20% testcases, $m = 1$ or $n = 1$ for the target matrix size;
- For 40% testcases, $1 \leq n_i < 10$ for the compressed String s;
- For 100% testcases, $1 \leq m, n \leq 200$ for the target matrix size and $1 \leq n_i \leq 100$ for the compressed String s.

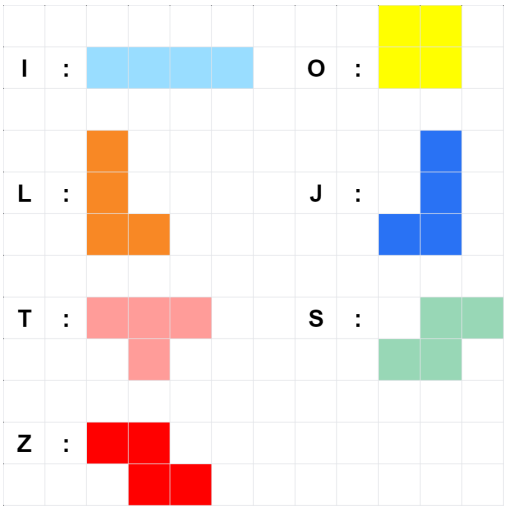
Problem 4. Trivial Tetris Game(Medium, 30 points)

Tetris is a traditional game. In this game, tetrominoes will fall from the sky and keep move downward until they touch the bottom of the field or land on a piece that had been placed before it.

In our trivial version of Tetris, there are several things that you should know:

- The tetromino will not be rotated.
- If one line is **completely filled**, the grids in this line will be eliminated. And the lines above this line will move down only one line (the grid will not freefall).
- If one tetromino **exceeds the upper bound** of the field **(touching the bound is acceptable)**, the game will be **ended immediately** and you should output the field at this time. You can take a look at Sample 3, which indicates this kind of scene.
- If a line is filled and the upper bound is exceeded at the same time, you should **first eliminate the line, then check the upper bound**. You can take a look at Sample 4, which indicates this kind of scene.

Below is the 7 types of tetrominoes in our trivial tetris game.



Input Format

First line gives the width w and height h of the field.

Second line gives the number of tetrominoes n.

The following n lines define t and s of each tetromino.

t means the type of the tetromino.

s means the index of the **leftmost grid** of the tetromino.

Output Format

The field with all tetrominoes correctly placed and eliminated.

Each grid of the field should be either 1 (filled) or 0 (unfilled).

Samples with Explanation

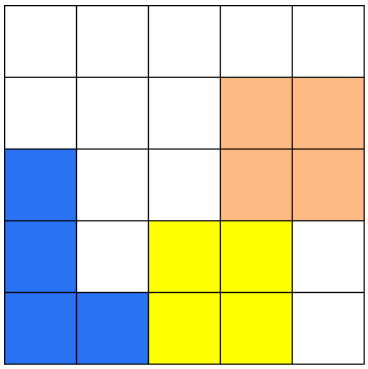
Sample 1

Input

```
5 5
3
L 0
O 2
O 3
```

output

```
00000
00011
10011
10110
11110
```



Explanation

The third tetromino falls on the second tetromino. And none of the lines is filled completely.

Sample 2

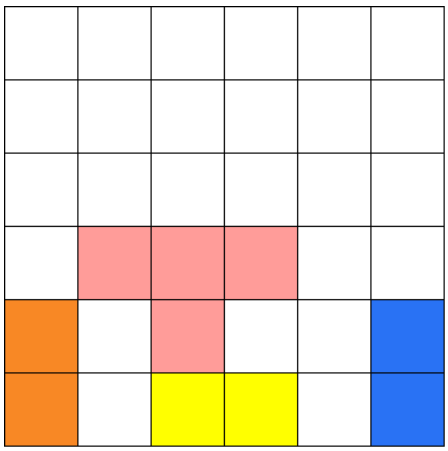
Input

```
6 6
4
L 0
O 2
J 4
T 1
```

Output

```
000000
000000
000000
011100
101001
101101
```

Explanation



After the third tetromino came, the bottom line is completely filled therefore eliminated. So, the final field will be like the above picture.

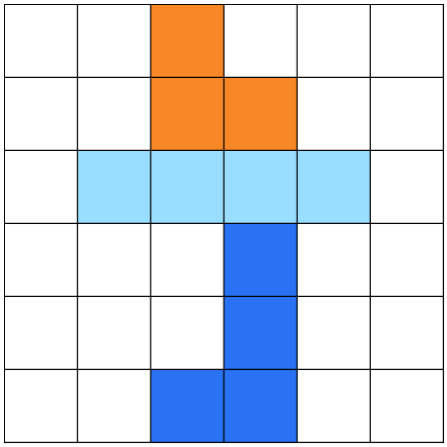
Sample 3

Input

```
6 6
3
J 2
I 1
L 2
```

Output

```
001000
001100
011110
000100
000100
001100
```



Explanation

The third tetromino exceeded the upper bound of the field, therefore the game was terminated.

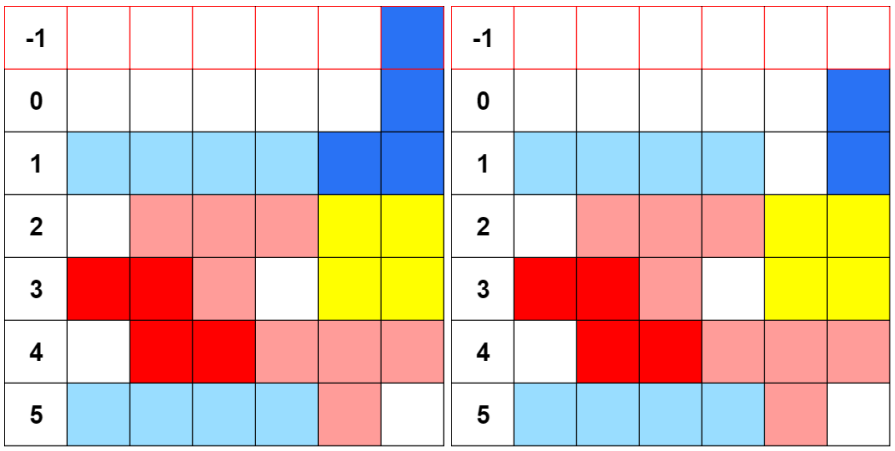
Sample 4

Input

```
6 6
8
I 0
T 3
Z 0
T 1
O 4
I 0
J 4
I 0
```

Output

```
000001
111101
011111
111011
011111
111110
```



Explanation

When the second to last tetromino came, it actually exceeded the upper bound of the field. However, line 1 was completely filled at the same time. Therefore line 1 was eliminated, and the game was not terminated. The final output should be like the bottom figure.

Hint

- You can create a **2-D array** to represent the field.
- In light of Sample 4, creating a **buffer area** of the field might be an assistance to you.
- For all testcases, $5 \leq w \leq 10, 3 \leq h \leq 10, 1 \leq n \leq 20$.
- For all testcases, t will be one of the 7 categories that listed above and t is a **CAPITAL** letter.
- For all testcases, s is always valid, which means the tetrominoes we provide will not exceed the left or right bound of the field.