Advanced 2048

Input file:	`stdin`
Output file:	`stdout`
Time limit:	2 seconds
Memory limit:	256 megabytes

Only lazy did not play the 2048 game last year. Soon, the secret of success had been revealed and interest to the game has faded. Nevertheless, the developers are now working on new modifications to the game. One of the prominent directions for the future development was the increased size of the game field and the increased final score.

However, in order to pick the optimal field size, it is required to emulate the gameplay many times on the fields of different sizes to estimate the number of moves needed to reach the goal under new conditions, and also to estimate the possible score values.

You are to write a program that will determine the final score given the description of a game session: field size, placement of the tiles and their values, the sequence of the player's moves and the description of all new tiles appearing on the field.

For those who are newbies to the popular game, the game rules are the following. 2048 is played on a $N \times N$ tile grid, with numbered tiles that all slide when a player moves them using the four arrow keys. After every turn, a new tile will randomly appear in an empty spot on the board with a value of either 2 or 4. Tiles slide as far as possible in the chosen direction until they are stopped by either another tile or the edge of the grid. If two tiles of the same number collide while moving, they will merge into a tile with the total value of the two tiles that collided. The resulting tile cannot merge with another tile again in the same move. The tile moves and mergers are resolved from the first tile to the last one (in terms of movement direction) consequently. For example, after moving to the right, the row "`2 2 _ 2 _`" turns into "`_ _ _ 2 4`", and the row "`2 2 2 2 2 2`" turns into "`_ _ _ 4 4 4`".

The player's score starts at zero, and is incremented whenever two tiles combine by the value of the new tile.

Input

The first line of input contains an integer N, the size of the field ($4 \le N \le 100$). The second line of input contains an integer K, the number of tiles already on the field ($0 \le K \le N^2$). In each of the next K lines, three integers are given to describe the tiles on the field: a number written in the tile (one of the powers of two from first (2) to tenth (1024)) and the tile coordinates on the field: row and column. Rows are numbered from one starting from the top, and columns are numbered from one starting from the left. It is guaranteed that the given tiles occupy K different cells of the grid.

After that, an integer L is given on a separate line: the number of player's moves in a game session ($0 \le L \le 10\,000$). The next L lines contain descriptions of the player's moves in their order, one description per line. A description of a move starts with one of the characters "`L`" (left), "`R`" (right), "`U`" (up) or "`D`" (down). Then follow three integers separated by single spaces. They define the new tile that appeared on the field after the player's move with the value of either 2 or 4 and the coordinates of the tile: row and column. It is guaranteed that the new tile is placed on an empty tile of the field after each player's move.

Output

Output a single integer: the final score in the game which is described in the input.

Examples

stdin	stdout
6 5 2 1 2 2 1 3 4 1 5 4 1 6 2 3 1 2 L 4 2 1 U 2 6 2	20
4 4 1024 1 1 1024 1 2 1024 2 1 1024 2 2 2 D 4 1 1 R 4 1 1	8192