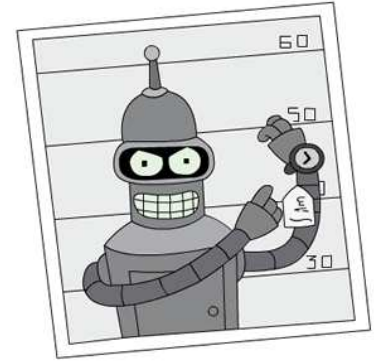


# Problem K

## Robotic Encryption

Due to suspicion of cheaters, one of the more paranoid problem setters has started encrypting all messages to the rest of the jury before sending them. He didn't use any standard encryption, however, as he believes those are all part of a giant conspiracy network trying to crush IDI Open from the inside. Instead, he based it on the fact that the cheaters are likely to be the worst programmers. The decryption requires some programming skill, and should therefore be safe.



Along with the encrypted message, he sent explanation of how to decrypt it. The only problem now is that not all jury members are able to implement the decryption. This is where we need your help. You need to help us decrypt these messages by writing a program that does the task.

Decryption is performed by simulating a robots movement on a grid. The robot is initially placed in the north-west corner of the grid, facing south. The robot is a simple one, and only accepts three different commands:

- L turns the robot 90° to the left.
- R turns the robot 90° to the right.
- F moves the robot one square forward. If moving forward would cause the robot to fall off the grid, the robot instead makes a 180° turn without moving.

Instructions to the robot is given in a series of commandsets. A commandset is a string of commands, with the possible addition of loops. A loop is given on the form "(commandset)number" where number is the number of times the commandset inside the parentheses should be run. Longer sequences of commands can be built up recursively in this fashion. More formally:

```
commandset ::= instruction+
instruction ::= command|loop
loop ::= "(" commandset ")" number
command ::= R | L | F
number ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
```

The decrypted text is the text string obtained by concatenating the characters on the grid positions the robot stands on after executing each commandline.

## Input specifications

The first line of input gives  $T$ , the number of test scenarios. Each scenario starts with a line containing  $W$  and  $H$ , separated by a single space, describing the dimensions of the grid. Then follows  $H$  lines, each consisting of  $W$  characters, making up the grid. After this comes a line containing  $N$ , the number of commandlines, followed by the  $N$  lines the robot will be executing.

## Output specifications

One line per test scenario, containing the decrypted text.

## Notes and Constraints

- $0 < T \leq 100$
- $0 < W \leq 50$
- $0 < H \leq 50$
- $0 < N \leq 20$
- Commandlines will be no longer than 50 characters, and will follow the syntax given in the problem text.
- No character with ASCII value lower than 32 or higher than 126 will appear on the robots grid.

## Sample input

```
1
6 7
012345
6789AB
CDEFGH
IJKLMN
OPQRST
UVWXYZ
_! .,&
12
FFL(F)5
(F)4
(LF)2
(L(R)6L)9
RFRFFF
(L(F)2)2
LF
FLFF
FFFF
LF
FLFF
L(F)4
```

## Output for sample input

```
HELLO WORLD!
```