Problem I
Karel the Robot
Time limit: 10 seconds

Did you know that the word "robot" is almost 100 years old? It was first introduced in 1920, in the science-fiction theatrical play *R.U.R.*, written by Karel Čapek. As a tribute to this Czech writer, an educational programming language was named Karel many years later at Stanford University. Your task is to implement an interpreter of a simplified version of this programming language.

The Karel programming language controls a robot named Karel, who lives in a grid of unit squares. Some of the squares are free, while others contain a barrier. Karel always occupies one of the free squares and faces one of the four cardinal directions. The two basic commands are "move forward" and "turn left." The language also provides simple conditional and looping statements. The main educational potential of the language lies in the possibility of defining new procedures for more complex tasks.

Our simplified version of the language can be described by the following grammar:

```
<program>    :=  "" | <command> <program>
<command>    :=  "m" | "l" | <proc-call> |
                 "i" <condition> "(" <program> ")(" <program> ")" |
                 "u" <condition> "(" <program> ")"
<condition>  :=  "b" | "n" | "s" | "e" | "w"
<proc-call>  :=  <uppercase-letter>
<proc-def>   :=  <uppercase-letter> "=" <program>
```

There are five types of commands:

- `m` ("move forward") advances Karel's position by one grid square in its current heading, unless there is a barrier, in which case the command has no effect.
- `l` ("turn left") makes Karel turn left 90 degrees.
- *X* where *X* is any uppercase letter, invokes the procedure named *X*.
- `i` ("if") followed by a single-letter condition, and two programs in parentheses. If the condition is satisfied, the first program is executed. Otherwise, the second program is executed.
- `u` ("until") followed by a single-letter condition, and a program in parentheses. If the condition is satisfied, nothing is done. Otherwise, the program is executed and then the command is repeated.

A condition can be either '`b`', which is satisfied if and only if there is a barrier in the next square in Karel's current heading, or one of the four directional letters '`n`', '`s`', '`e`', or '`w`', which is satisfied if and only if Karel's current heading is north, south, east, or west, respectively.

For instance, a simple program `ub(m)` can be understood to mean: "keep moving forward until there is a barrier," while `un(l)` means "turn to the north." A procedure definition `R=lll` defines a new procedure '`R`' which effectively means "turn right."

Input

The first line of input contains four integers r, c, d, and e, where r and c ($1 \leq r, c \leq 40$) are the dimensions of the grid in which Karel lives, d ($0 \leq d \leq 26$) is the number of procedure definitions, and e ($1 \leq e \leq 10$) is the number of programs to be executed.

Then follow r lines describing the grid (running north to south), each containing c characters (running west to east), each character being either '`.`' (denoting a free square) or '`#`' (denoting a barrier). All squares outside this given area are considered barriers, which means Karel may never leave the area.

Each of the next d lines contains a procedure definition, associating a procedure name (one uppercase letter) with a program forming the procedure body. No procedure name is defined more than once. Procedure bodies may contain invocations of procedures that have not yet been defined.

The last $2e$ lines describe the programs to be executed. Each such description consists of a pair of lines. The first line of each pair contains two integers i and j and a character h, where i ($1 \leq i \leq r$) is the row and j ($1 \leq j \leq c$) is the column of Karel's initial position, and $h \in \{\texttt{n}, \texttt{s}, \texttt{e}, \texttt{w}\}$ represents Karel's initial heading. It is guaranteed that the initial position is a free square. The second line of each pair contains a program to be executed from that initial position.

All procedure bodies and all programs to be executed are at least 1 and at most 100 characters long, syntactically correct, and only contain invocations of procedures that are defined. The lines with procedure definitions and programs to be executed contain no whitespace characters.

Output

For each program execution, output the final position of Karel after the complete program is executed from the respective initial position. Follow the format used to describe initial positions, that is, two numbers and a directional character. If a particular execution never terminates, output `inf` instead.

Sample Input 1

```
4 8 5 7
.......#
..#....#
.###...#
.....###
R=lll
G=ub(B)
B=ub(m)lib(l)(m)
H=ib()(mmHllmll)
I=III
1 1 w
G
1 1 e
G
2 2 n
G
2 6 w
BR
4 1 s
ib(lib()(mmm))(mmmm)
1 1 e
H
2 2 s
I
```

Sample Output 1

```
1 1 w
inf
1 1 w
2 4 s
4 4 e
1 4 e
inf
```