717 Calculating Expressions on Turing Machine

Turing Machine

Turing Machine (TM) was defined by mathematician Alan Turing in 1936. You probably expect all the Contest problems to be related to the same topic, so you may now wonder what the Turing Machine has in common with the Antique Theatre. The fact is that Alan Turing had a friend called Fred. And Fred's Grandmother was keen on Antique Tragedies. So we think it is a good idea to give you this problem as a remembrance of Alan Turing.

Let us describe one particular TM: TM consists of two-way potentially infinite *tape*, read/write *head* and a *finite automaton* control unit. The tape is an infinite one-dimensional sequence of *fields*. Each field contains one symbol of an *alphabet Sigma* $= \{\texttt{\~{}}, 0, 1, \ldots, M\}$, where '~' is a special *blank* symbol. At each moment, there is just a finite number of fields not containing the blank symbol.

The head is a device capable at each step of reading one symbol from the tape field above which it is positioned, writing another symbol on its place and moving one field to the left or right. As the tape is two-way potentially infinite, the movement is always possible.

The control unit drives the head. At each time, it is in one state taken from *Gamma* $= \{0, 1, ..., N\}$. It starts in state 0. At each step the control unit considers its actual state *gamma* and the symbol under the head *sigma*. This information determines the symbol to be written on the tape *sigma'* in place of *sigma*, the next state to go *gamma'* and the direction *delta* (`R` or `L` for right or left) for the movement of the head.

TM description (TMD) is a triple (M, N, P) where P is a set of rules. Each rule is a quintuple *(gamma, sigma, sigma', gamma', delta)* describing the behaviour of the machine in a particular situation as described in the preceding paragraph. If no rule exists for the current situation, the machine stops, i.e. the calculation is finished. Conflicting (with the same *gamma* and *sigma*) rules may not exist.

In the text form, TMD starts with a line containing positive integers M and N. Then there is an arbitrary number of lines containing each one rule. The rule is described as *gamma sigma sigma' gamma' delta*, where *gamma, sigma, sigma'* and *gamma'* are integers ('˜' should be coded as `-1` and *delta from {R, L}*, the symbols are separated by spaces. After the last rule, a line immediately follows which contains only the symbol '`-`'.

When the machine starts there will be a finite string of symbols from *Sigma* starting under the head position and continuing to the right. All the remaining fields on the tape are blank.

Theoretically, TM is equivalent to any general purpose computer. We ask you to at least partly demonstrate it — you are to write a program generating TMD evaluating arbitrary Turing arithmetic expression (TAE) for any input values. TAE is defined in the following section.

Turing Arithmetic Expression

TAE is defined by the following grammar:

- $TAE \rightarrow expr$
- $expr \rightarrow factor \mid expr + expr$
- $factor \rightarrow (\ expr\ ) \mid factor * factor \mid variable$
- $variable \rightarrow$ `1 | 2 | ... | 9`

The operators `+` resp. `*` operators stand for integer addition resp. multiplication modulo 10 (e.g. $238 * 17 = 6$); multiplication takes precedence over addition. The syntactic element *variable* denotes the value of the first, second, etc. up to ninth integer written on the tape of the TM at start time.

Task

Write a program that for each TAE outputs a TMD evaluating the TAE for any valid contents of the tape. A valid contents of the tape is a sequence of at most nine non-negative integers written left to right (most significant [non-zero] digit first) in a decimal notation using symbols `0`, ..., `9` and separated by one blank symbol. All the rest of the tape is blank, the head starts at the most significant digit of the first number. The magnitude of the integers is not specified. Example of a valid tape (underline indicates head position):

... ~ ~ $\underline{1}$ 2 3 ~ 4 7 ~ 1 1 ~ ~ ~ ~ ...

When the TM finished processing the tape should contain the result of the computation starting with the most significant digit under the head and continuing to the right until the first blank. No leading zeros are allowed. The contents of the rest of the tape is insignificant. For example, if the TAE was `(1+3)*2` and the tape contents as above, the correct answer would be $(123 + 11) * 47 \bmod 10 = 8$. The correct tape contents would be

```
x x x 8 ~ x x x
```

where x stands for any symbol.

Input

The input consists of N cases. The first line of the input contains only positive integer N. Then follow the cases. Each case is described by one line, containing one valid TAE. The line is at most 1000 characters long, does not contain any characters other than `0`, `1`, ..., `9`, `(`, `)`, `*` and `+`, and is terminated by the newline character.

Output

For each case, print valid TMD that evaluates the TAE for any valid tape contents. You can assume there will be always at least as many integers on the tape, as mentioned in the TAE.

Sample Input

```
2
1
2
```

Sample Output

```
9 2
0 1 1 1 R
0 2 2 1 R
0 3 3 1 R
0 4 4 1 R
0 5 5 1 R
0 6 6 1 R
0 7 7 1 R
0 8 8 1 R
0 9 9 1 R
```

```
1 0 0 2 L
1 1 1 2 L
1 2 2 2 L
1 3 3 2 L
1 4 4 2 L
1 5 5 2 L
1 6 6 2 L
1 7 7 2 L
1 8 8 2 L
1 9 9 2 L
1 -1 -1 2 L
-
9 1
0 0 0 0 R
0 1 1 0 R
0 2 2 0 R
0 3 3 0 R
0 4 4 0 R
0 5 5 0 R
0 6 6 0 R
0 7 7 0 R
0 8 8 0 R
0 9 9 0 R
0 -1 -1 1 R
-
```