Problem G. Shortest Accepted Word

Input file:	`shortest-accepted.in`
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

In this problem, we consider strings consisting of lowercase Latin letters "`a`", "`b`" and "`c`".

For this problem, let us define a *regular expression* recursively as follows:

1. A single character "`$`" is a regular expression accepting the empty string.
2. Single characters "`a`", "`b`" and "`c`" are regular expressions accepting strings "`a`", "`b`" and "`c`", respectively.
3. If P is a regular expression, "`(`P`)`" is also a regular expression accepting all strings accepted by P.
4. If P is a regular expression which is a **direct** result of applying any of the rules 1–4, its *iteration*, denoted as "P`*`", is also a regular expression accepting strings of the form $s = u_1 u_2 \dots u_k$ (a concatenation of zero or more strings) where k is any non-negative integer and each string u_i is accepted by P.
5. If P and Q are regular expressions which are **direct** results of applying any of the rules 1–5, their *concatenation*, denoted simply as "PQ", is also a regular expression accepting strings of the form $s = uv$ (a concatenation of u and v, each of which may be empty) where the prefix u is accepted by P and the suffix v is accepted by Q.
6. If P and Q are regular expressions which are **direct** results of applying any of the rules 1–6, their *union*, denoted as "P`|`Q", is also a regular expression accepting both strings accepted by P and strings accepted by Q.

The restrictions in rules 4–6 are imposed to prevent ambiguities and prioritize operations: when reading a regular expression, evaluate iteration, then concatenation, then union. Parentheses play the usual role of prioritizing operations enclosed in them. For example, the regular expression "`a(bac|ac*)`" is read as "accept `a` followed by either (`b` followed by `a` followed by `c`) or (`a` followed by zero or more copies of `c`)".

Given a regular expression r, find the shortest string s which is accepted by this regular expression r. If there are several such strings, find the lexicographicaly smallest one.

Input

The first line of input contains one integer T, the number of test cases ($1 \le T \le 300$).

Each of the next T lines describes a single test case. Each test case description consists of a regular expression r which is a string constructed by the above rules. Its length is from 1 to 300 characters.

The sum of all lengths of regular expressions is not greater than 300.

Output

For each test case print a single line containing the shortest string accepted by the given regular expression r. If there is more than one such string, print the **lexicographically smallest** one.

If the answer is an empty string, print a single character "`$`" instead.

Example

shortest-accepted.in	standard output
3 a ab|ac*(ca|cb) ((ab|ac)a)*	a ab $