The 27th Annual
ACM International Collegiate Programming Contest ASIA Regional - Taejon

Problem F
Digital Onion
Input: digion.in

“Digital Onion (DO, for short)” consists of parentheses. Null parenthesis (no parentheses) is a DO, and “( )” is also a DO. Then, all DOs can be formally defined recursively as follows:

(1) Definition: Null parenthesis is a DO. Especially, we call this the Null DO.
(2) Definition: “( )” is a DO. Especially, we call this the primitive DO.
(3) Definition: If both A and B are DOs, then the combined form of “(A) B” is also a DO, where we call A the inside of the DO and B the outside of the DO.

For example, “(( )) ( )” is a DO and “((( ))) (( ) ( ) ( )) ” is a DO, but “( ) ( )) ((” and “(( )))” cannot be a DO. Let us define the *weight* of a DO. The weight of a DO is defined as the number of '(' symbols, or equivalently the number of ')' symbols in it. Thus, the weight of the null DO is 0, and the weight of the primitive DO “( )” is 1. Also, it is easy to see that the weight of “((( ))) (( ) ( ) ( ))” is 7.

Also, if X = ((( ))) (( ) ( ) ( )) , the inside of X is (( )), and the outside of X is (( ) ( ) ( )). When X = (( ) (( ))), the inside DO of X is ( ) (( )), and the outside DO of X should be the Null DO.

Then, we define the price order of all DO objects. The ordering rules are simple, so we can sort all DOs according to their prices. This means that we can always determine which one is the more expensive between two different DOs given.

[Rule1]: The more weight, the more expensive.
[Rule2]: If the weights of two DOs are equal, then the price depends on the inside DO of the two DOs.
[Rule3]: If the weights of two DOs are the same and the prices of the inside DOs are equal, then the price depends on the outside DO of the two DOs.

Let us explain these rules. There are two DOs, A and B. We denote A < B to show that B is more expensive than A. It is easy to see ( ) < (( )) and (( )) ( ) < ((( ))) ( ) by Rule 1, (( )) (( )) < ((( ))) ( ) by Rule 2, and (( )) ( ) ( ) < (( )) (( )) by Rule 3.

Given a DO X, your task is to write the “Next More Expensive DO (NMED)” which is a more expensive DO than X, and there is no DO whose price is between NMED and X. That means that NMED is the next rank DO to a given DO when we sort all possible DOs according to their prices.

Input
The input consists of T test cases. The number of test cases (T) is given in the first line of the input file. Each test case DO is written in a single line. ‘$’ is placed at the end of each line to denote the end of each input DO representation, and there is at least one blank space to separate ‘(’s and ‘)’s and ‘$’. Note that the minimum weight of input DO is 1, and the maximum weight is 30.

Output

Print exactly one line for each test case. The line should contain the Next More Expensive DO (NMED) with an ending mark ‘$’. As was specified in the input format, there is at least one blank space to separate ‘(’s and ‘)’s and ‘$’.

Sample Input (digion.in)	**Output for the Sample Input**
`3`	`( ) ( ) $`
`( ) $`	`( ( ) ) $`
`( ) ( ) $`	`( ( ) ( ( ) ) ) ( ( ( ) ) ) $`
`( ( ) ( ( ) ) ) ( ( )( ) ) $`	