A. And Yet Another Bracket Sequence

time limit per test: 5 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

Polycarpus has a finite sequence of opening and closing brackets. In order not to fall asleep in a lecture, Polycarpus is having fun with his sequence. He is able to perform two operations:

- adding any bracket in any position (in the beginning, the end, or between any two existing brackets);
- cyclic shift — moving the last bracket from the end of the sequence to the beginning.

Polycarpus can apply any number of operations to his sequence and adding a cyclic shift in any order. As a result, he wants to get the correct bracket sequence of the minimum possible length. If there are several such sequences, Polycarpus is interested in the lexicographically smallest one. Help him find such a sequence.

A*correct bracket sequence* is a sequence of opening and closing brackets, from which you can get a correct arithmetic expression by adding characters "1" and "+" . Each opening bracket must correspond to a closed one. For example, the sequences "(())()", "()", "(()(()))" are correct and ")(", "(()" and "(()))(" are not.

The sequence a_1 $a_2 ...$ a_n is lexicographically smaller than sequence b_1 $b_2 ...$ b_n, if there is such number i from 1 to n, that$a_k = b_k$ for $1 \le k < i$ and $a_i < b_i$. Consider that "(" $<$ ")".

Input

The first line contains Polycarpus's sequence consisting of characters "(" and ")". The length of a line is from 1 to $1\,000\,000$.

Output

Print a correct bracket sequence of the minimum length that Polycarpus can obtain by his operations. If there are multiple such sequences, print the lexicographically minimum one.

Examples

input

```
()(())
```

output

```
(())()
```

input

```
()(
```

output

```
(())
```

Note

The sequence in the first example is already correct, but to get the lexicographically minimum answer, you need to perform four cyclic shift operations. In the second example you need to add a closing parenthesis between the second and third brackets and make a cyclic shift. You can first make the shift, and then add the bracket at the end.