C. Bracket Sequence

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

A *bracket sequence* is a string, containing only characters "(", ")", "[" and "]".

A *correct bracket sequence* is a bracket sequence that can be transformed into a correct arithmetic expression by inserting characters "1" and "+" between the original characters of the sequence. For example, bracket sequences "()[]", "([])" are correct (the resulting expressions are: "(1)+[1]", "([1+1]+1)"), and "](" and "[" are not. **The empty string is a correct bracket sequence by definition.**

A *substring* $s[l... r]$ $(1 \le l \le r \le |s|)$ of string $s = s_1 s_2 ... s_{|s|}$ (where $|s|$ is the length of string s) is the string $s_l s_{l+1} ... s_r$. **The empty string is a substring of any string by definition**.

You are given a bracket sequence, not necessarily correct. Find its substring which is a correct bracket sequence and contains as many opening square brackets «[» as possible.

Input

The first and the only line contains the bracket sequence as a string, consisting only of characters "(", ")", "[" and "]". It is guaranteed that the string is non-empty and its length doesn't exceed 10^5 characters.

Output

In the first line print a single integer — the number of brackets «[» in the required bracket sequence. In the second line print the optimal sequence. If there are more than one optimal solutions print any of them.

Examples

input
([])
output
1 ([])

input
(((
output
0