Longest Correct Expression

140 points

Memory Limit: 32 MB

Time Limit: 2 seconds

Consider the following description that defines “correct expressions of brackets”.

Basis Clauses:

- () is a correct expression.
- {} is a correct expression.
- [] is a correct expression.

Inductive Clauses:

- If E is any correct expression, then (E) is a correct expression.
- If E is any correct expression, then {E} is a correct expression.
- If E is any correct expression, then [E] is a correct expression.
- If E and F are correction expressions, then EF is a correction expression.

Restriction Clause:

- No other expressions can be correct expressions other than those derived from the basis clauses and the inductive clauses.

For example, followings are some correct expressions of brackets according to the above definitions:

- ({[]})
- (){}[]
- [{[({})]}]{(){}({[[]]}){[]}}

And followings are some examples of expressions that are *not* correct:

- (
- (}]
- {[(}(]

Note that according to the definition, there cannot be any white space between any two symbols in a correct expression since a white space is not a symbol used in the definition.

Write a program to find the length of the longest correct subexpression in a given string consisting of following six characters: (,), [,], { and }. You also need to find the number of such longest subexpressions and print all the correct subexpressions.

INPUT

Input consists of a single line containing a string consisting of one or more of the six characters defined in the problem description. The length of the input string is guaranteed to be greater than or equal to 1 and less than or equal to 1000000.

OUTPUT

The first line of the output contains two integer values, *l* and *c*, representing the length of the longest substring of a correct expression of brackets, and the number of such longest subexpressions, respectively. Next *c* lines of the output will list such correct expressions each one on a line, *in the order of their appearance* (from left to right) in the input string with the left most one appearing on the second line of the output and the right most one on the last line.

When the input string contains no substring of correct expression of brackets, you must output only one line containing 0 0.

Sample Inputs/Outputs

Input	Output
`{[(}(]`	`0 0`

`(){}[]`	`6 1`
	`(){}[]`

`)[{()}])([]()){ )[]{`	`6 2`
	`[{()}]`
	`([]())`