Given some pairs of corresponding open and close parenthesis, we can use them to build an infinite number of correct parenthesizations. For instance, with the pairs \( ( ) \) and \( [ ] \), all correct parenthesizations are defined by the grammar

\[
P \to < \text{empty word} > \\
P \to (P) P \\
P \to [P] P
\]

Can you generate all correct parenthesizations of a given size?

**Input**

Input consists of a non-empty string \( s \) and a strictly positive even number \( n \). The string \( s \) has even size, and includes the corresponding pairs of open and close parenthesis: \( s[0] \) with \( s[1], s[2] \) with \( s[3] \), etc.

**Output**

Print all correct parenthesizations of size \( n \) that can be made up with the corresponding open and close parenthesis included in \( s \).

**Observation**

You can print the parenthesizations in any order.

**Sample input 1**

\[
( ) 6
\]

**Sample output 1**

\[
( ) ( ) \\
( ( )) \\
( ( ( )))
\]

**Sample input 2**

\[
( ) ( [ ] ) 2
\]

**Sample output 2**

\[
( ) \\
()
\]

**Sample input 3**

\[
[ ] ( ) 4
\]

**Sample output 3**

\[
[ ] [ ] \\
( ) ( [ ] ) \\
( ) ( ) \\
( [ ] ) \\
( [ ] ) \\
( ) \\
()
\]