Splitting Parentheses

A **parentheses string** is a string consisting only of (and).

A parentheses string is called **good** if and only if it can be constructed using the following rules:

- The empty string is good.
- If X is good, then (X) is good.
- If X and Y are good, then XY is good.

No other parentheses string is good.

Kiki is given a good parentheses string S of length N. She must cut S at $K-1$ positions into exactly K nonempty consecutive pieces $P_1, P_2, \ldots, P_K$.

After Kiki makes the cuts, Mog chooses a permutation of the K pieces and concatenates them to form a string T. Mog then repeatedly removes an adjacent pair $()$ from T until no such pair remains. The **score** is the length of the remaining string.

For a fixed partition, its **value** is the maximum score that Mog can obtain by rearranging the pieces. Kiki wants to find a partition with the smallest possible value.

Let X be this smallest value. Output X and any partition of S whose value is X.

Input

The first line contains two integers N and K, the length of S and the number of pieces.

The second line contains the string S.

Output

On the first line, output an integer X, the smallest possible value of a partition.

On each of the next K lines, output one of the pieces $P_1, P_2, \ldots, P_K$, in their original order.

Each piece must be nonempty, and their concatenation must be S. The value of the output partition must be exactly X.

If there are several optimal partitions, output any of them.

Constraints

- $2 \leq N \leq 300$.
- N is even.
- $1 \leq K \leq N$.
- S has length N and consists only of $($ and $)$.
- S is a good parentheses string.

Subtasks and Scoring

Subtask	Score	Additional Constraints
1	6	$N \leq 10$
2	2	$K = 1$
3	7	$K \leq 3$
4	15	The first $N/2$ characters of S are $($, and the remaining $N/2$ characters are $)$.
5	33	$N \leq 70$
6	37	No additional constraints.

In each subtask, you can obtain 50% of the subtask score by outputting the correct value X, even if the partition is not correct.

More precisely, for each test case, your output:

- gets full points if X is correct and the following K lines describe a valid partition whose value is X;
- gets 50% of the points if X is correct but the following K lines do not describe such a partition;
- gets 0 points otherwise.

The score for each subtask is the minimum of the points for the test cases in the subtask.

Examples

Sample Input 1

```
8 4
(()(()))
```

Sample Output 1

```
2
(
()
(())
)
```

The last four lines divide S into the pieces $($, $()$, $(())$, and $)$.

Mog can arrange them in the order $)$, $()$, $(())$, $($. After all possible removals, $)($ remains, so the score is 2. No ordering of these pieces gives a score greater than 2, and no partition into four pieces has a value smaller than 2.

Sample Input 2

```
4 2
()()
```

Sample Output 2

```
0
()
()
```

Sample Input 3

```
12 6
((((((((((((
```

Sample Output 3

```
6
(
(
(
((()))
)
))
```