F. Paper task

time limit per test: 3 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Alex was programming while Valentina (his toddler daughter) got there and started asking many questions about the round brackets (or parenthesis) in the code. He explained her a bit and when she got it he gave her a task in order to finish his code on time.

For the purpose of this problem we consider only strings consisting of opening and closing round brackets, that is characters '`(`' and '`)`'.

The sequence of brackets is called *correct* if:

1. it's empty;
2. it's a correct sequence of brackets, enclosed in a pair of opening and closing brackets;
3. it's a concatenation of two correct sequences of brackets.

For example, the sequences "`()()`" and "`((()))(())`" are correct, while "`)(()`", "`(((((`" and "`())`" are not.

Alex took a piece of paper, wrote a string s consisting of brackets and asked Valentina to count the number of **distinct** non-empty substrings of s that are correct sequences of brackets. In other words, her task is to count the number of non-empty correct sequences of brackets that occur in a string s as a **substring** (don't mix up with subsequences).

When Valentina finished the task, Alex noticed he doesn't know the answer. Help him don't loose face in front of Valentina and solve the problem!

Input

The first line of the input contains an integer n ($1 \le n \le 500\,000$) — the length of the string s.

The second line contains a string s of length n consisting of only '`(`' and '`)`'.

Output

Print the number of **distinct** non-empty correct sequences that occur in s as substring.

Examples

input

```
10
()()()()()
```

output

```
5
```

input

```
7
)(())()
```

output

```
3
```

Note

In the first sample, there are 5 distinct substrings we should count: "`()`", "`()()`", "`()()()`", "`()()()()`" and "`()()()()()`".

In the second sample, there are 3 distinct substrings we should count: "`()`", "`(())`" and "`(())()`".