Problem A
Indivisible Bracket

Given a string S consisting of only '(' and ')', find the length of the longest substring of S that is an indivisible balanced bracket sequence.

A balanced bracket sequence is a bracket sequence that satisfies the following constraints.

- an empty string is a balanced bracket sequence
- if *a* is a balanced bracket sequence, then so is (*a*)
- if *a* and *b* are balanced bracket sequences, then so is *ab*

For example, `(()())` and `()(())` are balanced bracket sequences, but `(()(` and `())()` are not.

A balanced bracket sequence R of length m is indivisible if and only if there is no i such that $1 \leq i < m$ and both $R_{1..i}$ and $R_{i+1..m}$ are balanced bracket sequences. In other words, R cannot be split into two strings such that both resulting strings are balanced bracket sequences.

For example, `(()())` and `((()))` are indivisible, but `(())()` is not. Observe that `(())()` can be split into `(())` and `()`, and both are balanced bracket sequences.

Input

Input begins with an integer T ($1 \leq T \leq 1000$) representing the number of cases.

Each case contains a string S that consists of only character '(' or ')'. The length of S is at least 1 and does not exceed $100\,000$.

It is guaranteed that the total length of S over all cases does not exceed 10^6.

Output

For each case, output in a line "`Case #X: Y`" (without quotes) where `X` is the case number (starts from 1) and `Y` is the output for the respective case.

Sample Input #1

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

Sample Output #1

```
Case #1: 6
Case #2: 4
Case #3: 0
Case #4: 2
```

Explanation for the sample input/output #1

For the 1^{st} case, the longest indivisible balanced bracket sequence is `(()())` of length 6.

For the 2^{nd} case, the longest indivisible balanced bracket sequence is `(())` of length 4.

For the 3^{rd} case, the longest indivisible balanced bracket sequence is an empty string of length 0.