Daily Coding Problem #86

Problem

This problem was asked by Google.

Given a string of parentheses, write a function to compute the minimum number of parentheses to be removed to make the string valid (i.e. each open parenthesis is eventually closed).

For example, given the string "()())()", you should return 1. Given the string ")(", you should return 2, since we must remove all of them.

Solution

For a string to be considered valid, each open parenthesis should eventually be closed. Other parentheses that don't satisfy this condition should be counted as invalid.

Whenever we encounter an unmatched closing parenthesis, we can count it as invalid. After that, we also add the number of unmatched opening parentheses to our invalid count. This runs in O(N).

```
def count_invalid_parenthesis(string):
    opened = 0
    invalid = 0
    for c in string:
        if c == '(':
            opened += 1
        elif c == ')':

            if opened > 0:
```

```
                opened -= 1
            else:
                invalid += 1
    # Count as invalid all unclosed parenthesis
    invalid += opened
    return invalid
```