Minimum Add To Make Parentheses Valid

solved by	Senan
Platform	LeetCode
difficulty	Medium
# Serial	921
tags	String Manipulation
language	C++
solved on	@09/10/2024
link	https://leetcode.com/problems/minimum-add-to-make-parentheses-valid/description/
☑ Completion	☑

Intuition

The problem asks us to determine the minimum number of parentheses we need to add to make the string of parentheses valid. A valid parentheses string means every opening parenthesis `(` has a corresponding closing parenthesis `)`.

The approach is based on counting the number of unbalanced open and close parentheses. Every time we encounter an unmatched `)`, we increment the count of required closing parentheses, and similarly for unmatched `(`, we increment the count of required opening parentheses.

Approach

- Initialize two counters: `open` to track unmatched opening parentheses `(` and `close` to track unmatched closing parentheses `)`.
- Traverse the string:
  - If the character is `(`, increment the `open` counter.
  - If the character is `)` and there is an unmatched `(`, decrement the `open` counter because we can now match this `)` with an earlier `(`.
  - If the character is `)` and there is no unmatched `(`, increment the `close` counter because this `)` is unmatched and we need an extra `(`.
- After the loop, the sum of `open` and `close` will give the number of parentheses needed to make the string valid.

Complexity

Time Complexity:

The time complexity is **O(n)**, where `n` is the length of the string `s`. This is because we only traverse the string once.

Space Complexity:

The space complexity is **O(1)** as we are using a constant amount of extra space for the counters `open` and `close`.

Code

```
class Solution {
public:
    int minAddToMakeValid(string s) {
        int open = 0;
        int close = 0;

        for (auto &ch : s) {
            if (ch == '(') {
                open++;
            } else {
                if (open > 0) open--;
                else close++;
            }
        }
        return open + close;
    }
};
```