921. Minimum Add to Make Parentheses Valid

A parentheses string is valid if and only if:

- It is the empty string,
- It can be written as `AB` (`A` concatenated with `B`), where `A` and `B` are valid strings, or
- It can be written as `(A)`, where `A` is a valid string.

You are given a parentheses string `s`. In one move, you can insert a parenthesis at any position of the string.

- For example, if `s = "()))"`, you can insert an opening parenthesis to be `"(()))"` or a closing parenthesis to be `"())))"`.

Return *the minimum number of moves required to make* `s` *valid*.

Example 1:

```
Input: s = "())"
Output: 1
```

Example 2:

```
Input: s = "((("
Output: 3
```

Example 3:

```
Input: s = "()"
Output: 0
```

Example 4:

```
Input: s = "()))(("
Output: 4
```

Constraints:

- `1 <= s.length <= 1000`
- `s[i]` is either `'('` or `')'`.

```
class Solution:
    def minAddToMakeValid(self, s: str) -> int:
        if len(s)==0:
```

```
            return 0

        stack = []

        for i in range(len(s)):
            if s[i]==')':
                if len(stack) and stack[-1]=='(':
                    stack.pop()
                else:
                    stack.append(s[i])
            else:
                stack.append(s[i])

        return len(stack)
```