□ LeetCode 1021 – Remove Outermost Parentheses

◈ Problem Statement (Simple Words)

You are given a **valid parentheses string** s.

A **primitive parentheses string** is one that:

- Is non-empty
- Is valid
- Cannot be split into two valid parentheses strings

☞ For **each primitive**, remove its **outermost parentheses**
☞ Finally, **combine all results**

◈ Example

Input

s = "(()())(())"

Step 1: Split into primitives

"(()())" "(())"

Step 2: Remove outermost parentheses

"()()" "()"

Output

"()()()"

□ Key Idea (Using COUNT variable)

Instead of using a stack, we use a **counter**:

- count tracks **how deep** we are inside parentheses
- '(' → increase count
- ')' → decrease count

🔑 Important Rule

Character	When to ADD to result
(	if count > 0
)	if count > 1

Why?

- The **outermost parentheses** always occur when:
  - '(' → count == 0
  - ')' → count == 1
- We **skip those**

🔍 **Dry Run (Step-by-Step)**

Input

s = "(()())(())"

Initialize

count = 0

result = ""

Walk Through Each Character

Char	Before count	Action	After count	Add to result?	Result
(	0	count++	1	❌ outermost	""
(	1	count++	2	✅	"("
)	2	add	count--	1	"()"
(	1	count++	2	✅	"()("
)	2	add	count--	1	"()()"
)	1	count--	0	❌ outermost	"()()"
(	0	count++	1	❌	"()()"
(	1	count++	2	✅	"()()("
)	2	add	count--	1	"()()()"
)	1	count--	0	❌	"()()()"

✅ **Final Output**

"()()()"

□ Why This Works

- count == 0 → 1 → entering a primitive → **skip '('**
- count == 1 → 0 → exiting a primitive → **skip ')'**
- Everything inside → **keep it**

✓ Code Implementations

Java Code

```
class Solution {
    public String removeOuterParentheses(String s) {
        StringBuilder result = new StringBuilder();
        int count = 0;

        for (char ch : s.toCharArray()) {
            if (ch == '(') {
                if (count > 0) {
                    result.append(ch);
                }
                count++;
            } else { // ')'
                if (count > 1) {
                    result.append(ch);
                }
                count--;
            }
        }
        return result.toString();
    }
}
```

C++ Code

```
class Solution {
public:
    string removeOuterParentheses(string s) {
        string result = "";
        int count = 0;

        for (char ch : s) {
            if (ch == '(') {
                if (count > 0)
                    result += ch;
                count++;
            } else {
                if (count > 1)
                    result += ch;
                count--;
            }
        }
        return result;
    }
};
```

Python Code

```
class Solution:
    def removeOuterParentheses(self, s: str) -> str:
        result = []
        count = 0

        for ch in s:
            if ch == '(':
                if count > 0:
                    result.append(ch)
```

```
            count += 1
        else:
            if count > 1:
                result.append(ch)
            count -= 1

    return "".join(result)
```

⏱ **Complexity Analysis**

Metric	Value
Time	**O(n)**
Space	**O(n)** (output only)