删除最外层的括号

有效括号字符串为空 (""、"(" + A + ")" 或 A + B，其中 A 和 B 都是有效的括号字符串，+ 代表字符串的连接。例如，""，"()"，"(()())()" 和 "(()(()))" 都是有效的括号字符串。

如果有效字符串 S 非空，且不存在将其拆分为 S = A+B 的方法，我们称其为**原语（primitive）**，其中 A 和 B 都是非空有效括号字符串。

给出一个非空有效字符串 S，考虑将其进行原语化分解，使得：S = P_1 + P_2 + ... + P_k，其中 P_i 是有效括号字符串原语。

对 S 进行原语化分解，删除分解中每个原语字符串的最外层括号，返回 S 。

示例 1：

输入："(()())(())"

输出："()()()"

解释：

输入字符串为 "(()())(())"，原语化分解得到 "(()())" + "(())"，

删除每个部分中的最外层括号后得到 "()()" + "()" = "()()()"。

示例 2：

输入："(()())(())(()(()))"

输出："()()()()(())"

解释：

输入字符串为 "(()())(())(()(()))"，原语化分解得到 "(()())" + "(())" + "(()(()))"，

删除每个部分中的最外层括号后得到 "()()" + "()" + "()(())" = "()()()()(())"。

示例 3：

输入："()()"

输出：""

解释：

输入字符串为 "()()"，原语化分解得到 "()" + "()"，

删除每个部分中的最外层括号后得到 "" + "" = ""。

```
class Solution {
public:
    string removeOuterParentheses(string s) {
//左括号入栈，遇到右括号则将左括号出栈。外层括号不加入结果中
//怎么判断是外层括号？1.c 为左括号且入栈前栈为空 2.c 为右括号且栈顶左括号出栈后栈为空
        string res="";
        stack<char> st;
        for(auto& ch:s)
        {
            if(ch=='(')
            {
                if(!st.empty())
                {
                    res+=ch;
                }
                st.push(ch);
            }
            else
            {
                st.pop();
                if(!st.empty())
                {
                    res+=ch;
                }
            }
        }
        return res;
    }
};
```