

HW_Remove Outermost Parentheses 4

A valid parentheses string is either empty "", "(A + B)", or A + B, where A and B are valid parentheses strings, and + represents string concatenation.

Return s after removing the outermost parentheses of every primitive string in the primitive decomposition of s.

Input Format

The first line be String S.

Constraints

1 <= s.length <= 10^5

s[i] is either '(' or ')'.

s is a valid parentheses string.

Output Format

Return s after removing the outermost parentheses of every primitive string in the primitive decomposition of s.

Sample Input 0

((()))()

Sample Output 0

()()

Handwritten notes and diagrams illustrating the algorithm logic:

- Diagram of a stack structure with a horizontal line and a vertical line, containing the text "if push string" and "else pop".
- Diagram showing the string "((()))()" with the outermost parentheses removed, resulting in "()()".
- Diagram showing the string "((()))()" with the outermost parentheses removed, resulting in "()()".
- Diagram showing the string "((()))()" with the outermost parentheses removed, resulting in "()()".

Handwritten code snippets:

```
if ( '(' ) {  
    if ( !st > 0 )  
        st = 0;  
    st.push(c);  
} else {  
    if ( !st > 0 )  
        st = 0;  
    st.pop();  
}
```

Handwritten diagrams illustrating the algorithm logic:

- Diagram showing a stack structure with a horizontal line and a vertical line, containing the text "if push string" and "else pop".
- Diagram showing the string "((()))()" with the outermost parentheses removed, resulting in "()()".
- Diagram showing the string "((()))()" with the outermost parentheses removed, resulting in "()()".

Submitted Code

```
Language: Java 15  
1 import java.io.*;  
2 import java.util.*;  
3  
4 public class Solution {  
5  
6     public static void main(String[] args) {  
7         Scanner sc = new Scanner(System.in);  
8         String str = sc.nextLine();  
9         Stack<Character> st = new Stack<>();  
10        String ans = "";  
11  
12        for (int i=0; i<str.length(); i++){  
13            char ch = str.charAt(i);  
14            if (ch == '(') {  
15                if (st.isEmpty()) ans += ch;  
16                st.push(ch);  
17            } else {  
18                if (st.size() > 1) ans += ch;  
19                st.pop();  
20            }  
21        }  
22        System.out.print(ans);  
23    }  
24 }  
25 }
```

Handwritten diagrams illustrating the algorithm logic:

- Diagram showing a stack structure with a horizontal line and a vertical line, containing the text "if push string" and "else pop".
- Diagram showing the string "((()))()" with the outermost parentheses removed, resulting in "()()".
- Diagram showing the string "((()))()" with the outermost parentheses removed, resulting in "()()".