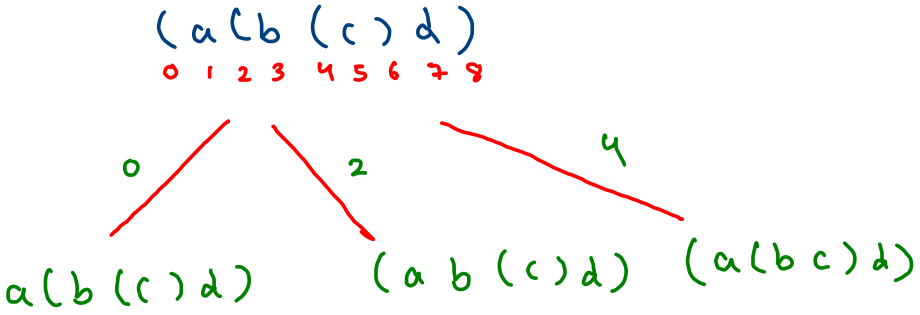
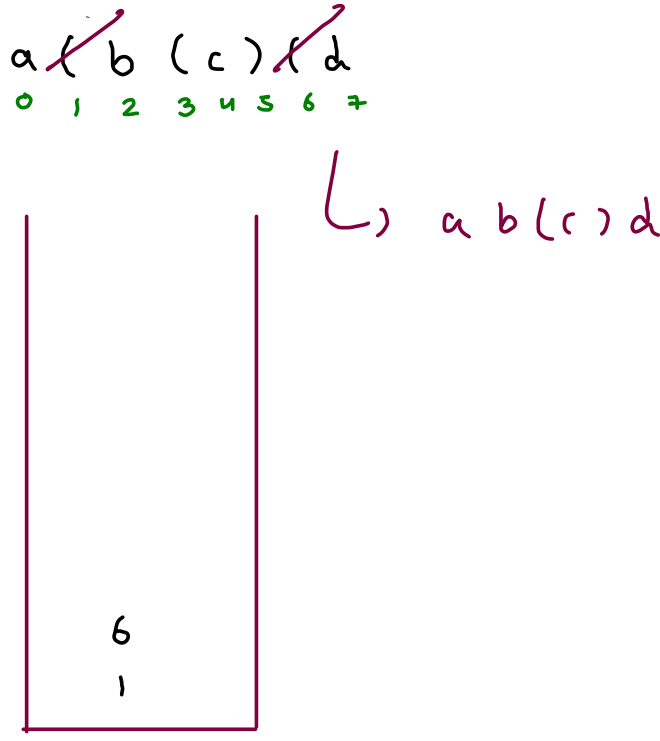
1249. Minimum Remove to Make Valid Parentheses

example
(a(b(c)d) this string has (a(bc)d), (ab(c)d) and a(b(c)d) 3 valid strings.
Among all 3 valid strings a(b(c)d) has the innermost parentheses.

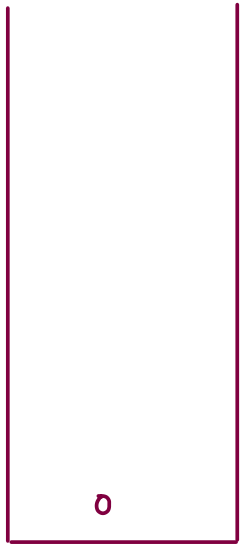




(a(b(c)d)

0 1 2 3 4 5 6 7 8

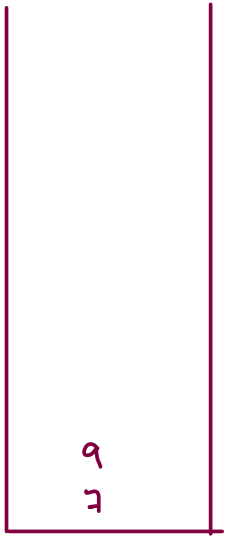
↳ a(b(c)d)



(a(b))c)d)

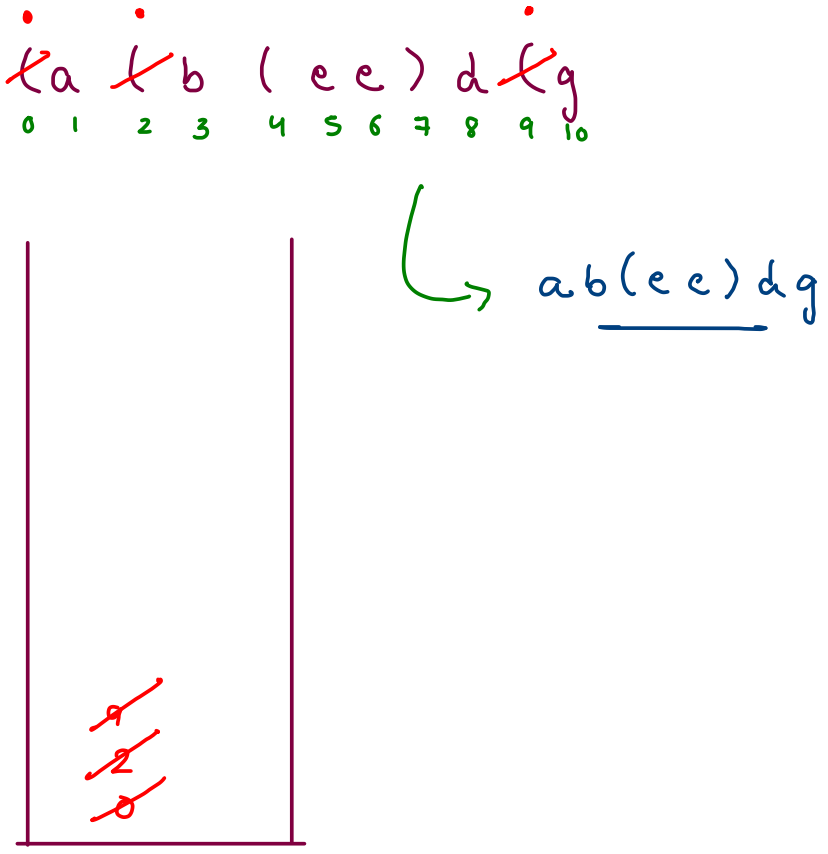
0 1 2 3 4 5 6 7 8 9

↳ (a(b))c



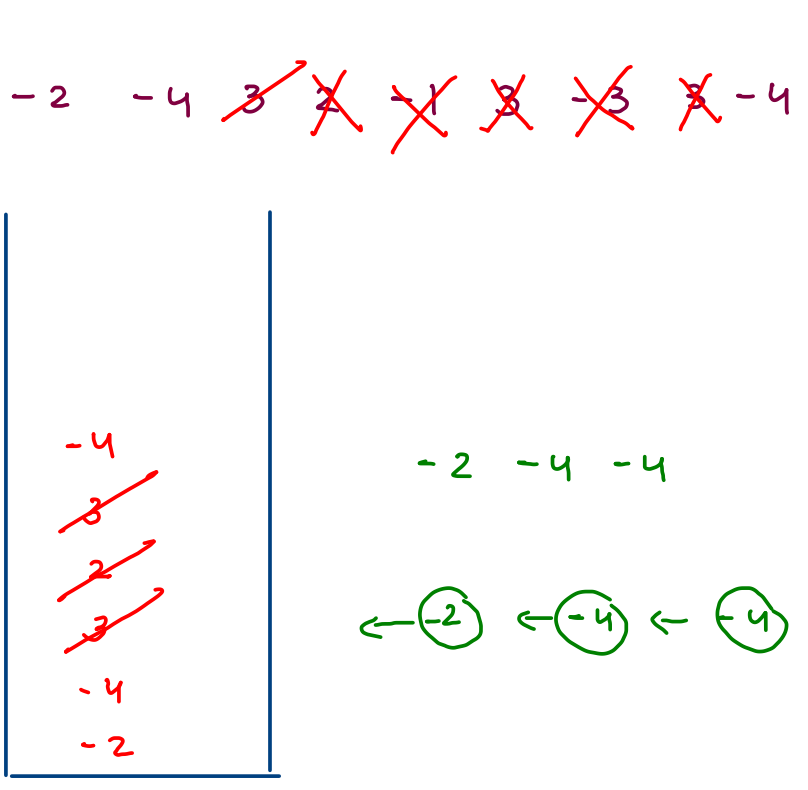
```
for(int i=0; i < s.length();i++) {
    char ch = s.charAt(i);

    if(ch == '(') {
        st.push(i);
    }
    else if(ch == ')'){
        if(st.size() > 0  && s.charAt(st.peek()) == '(') {
            st.pop();
        }
        else {
            st.push(i);
        }
    }
}
```

```
char[]arr = s.toCharArray();

while(st.size() > 0) {
    int idx = st.pop();
    arr[idx] = '.';
}

StringBuilder sb = new StringBuilder();

for(int i=0; i < arr.length;i++) {
    if(arr[i] != '.') {
        sb.append(arr[i]);
    }
}

return sb.toString();
```



735. Asteroid Collision

```
for(int i=0; i < asteroids.length;i++) {
    int val = asteroids[i];

    if(st.size() > 0 && st.peek() > 0 && val < 0) {
        while(st.size() > 0 && st.peek() > 0 && st.peek() < -val) {
            st.pop();
        }
        if(st.size() > 0 && st.peek() == -val) {
            st.pop();
        }
        else if(st.size() > 0 && st.peek() > -val){
            continue;
        }
        else {
            st.push(val);
        }
    }
    else {
        st.push(val);
    }
}
```



1003. Check If Word Is Valid After Substitutions

Medium 527 429 Add to List Share

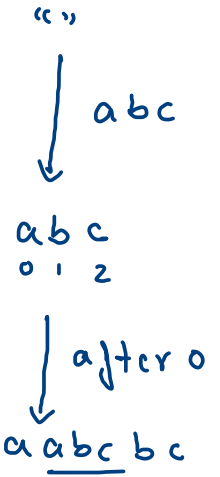
Given a string `s`, determine if it is **valid**.

A string `s` is **valid** if, starting with an empty string `t = ""`, you can **transform** `t` **into** `s` after performing the following operation **any number of times**:

- Insert string `"abc"` into any position in `t`. More formally, `t` becomes t_{left} + `"abc"` + t_{right}, where `t == `t_{left} + t_{right}. Note that t_{left} and t_{right} may be **empty**.

Return `true` *if* `s` *is a* ***valid*** *string, otherwise, return* `false`.

a abcbc



Example 1:

```
Input: s = "aabcbc"
Output: true
Explanation:
"" -> "abc" -> "aabcbc"
Thus, "aabcbc" is valid.
```

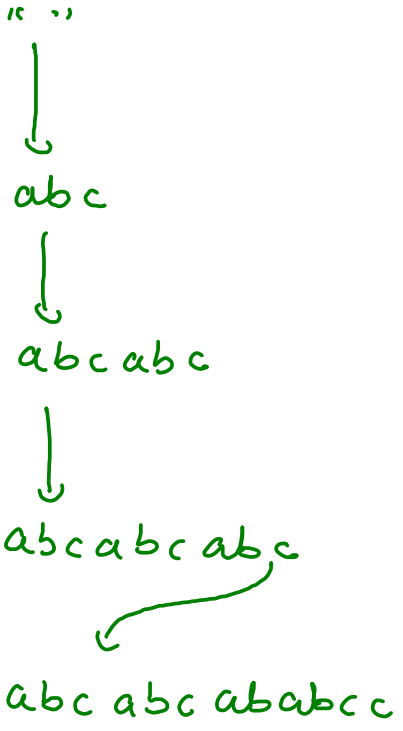
Example 2:

```
Input: s = "abcabcababcc"
Output: true
Explanation:
"" -> "abc" -> "abcabc" -> "abcabcabc" -> "abcabcababcc"
Thus, "abcabcababcc" is valid.
```

Example 3:

```
Input: s = "abccba"
Output: false
Explanation: It is impossible to get "abccba" using the operation.
```

abc abc aba b cc



s = abccba