宫水三叶的刷题日记

队列

Author ：宫水三叶
Date ：2021/10/07
QQ Group: 703311589
WeChat ：oaoaya

宫水三叶の刷题日记
公众号：宫水三叶的刷题日记

> **🍭 更多精彩内容，欢迎关注：公众号 / Github / LeetCode / 知乎 **

噔噔噔噔，这是公众号「宫水三叶的刷题日记」的原创专题「队列」合集。

本合集更新时间为 2021-10-07，大概每 2-4 周会集中更新一次。关注公众号，后台回复「队列」即可获取最新下载链接。

💡**下面介绍使用本合集的最佳使用实践：**

学习算法：

1. 打开在线目录（Github 版 & Gitee 版）；
2. 从侧边栏的类别目录找到「队列」；
3. 按照「推荐指数」从大到小进行刷题，「推荐指数」相同，则按照「难度」从易到难进行刷题‘
4. 拿到题号之后，回到本合集进行检索。

维持熟练度：

1. 按照本合集「从上往下」进行刷题。

学习过程中遇到任何困难，欢迎加入「每日一题打卡 QQ 群：703311589」进行交流 🍭🍭🍭

> **🍭 更多精彩内容，欢迎关注：公众号 / Github / LeetCode / 知乎 **

题目描述

这是 LeetCode 上的 **1047. 删除字符串中的所有相邻重复项** ，难度为 **简单**。

Tag：「队列」、「模拟」

给出由小写字母组成的字符串 S，重复项删除操作会选择两个相邻且相同的字母，并删除它们。

在 S 上反复执行重复项删除操作，直到无法继续删除。

在完成所有重复项删除操作后返回最终的字符串。答案保证唯一。

示例：

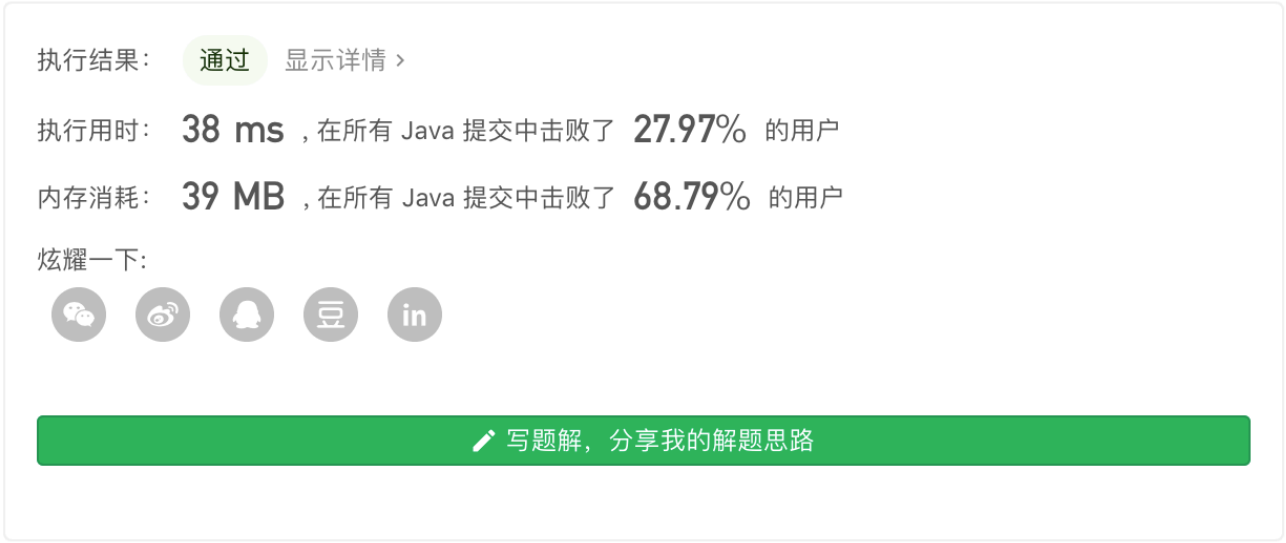
宫水三叶の刷题日记
公众号：宫水三叶的刷题日记

输入："abbaca"
输出："ca"
解释：
例如，在 "abbaca" 中，我们可以删除 "bb" 由于两字母相邻且相同，这是此时唯一可以执行删除操作的重复项。之后我们得到字符串 "aac

提示：

- 1 <= S.length <= 20000
- S 仅由小写英文字母组成。

(自带) 栈解法



宫水三叶の刷题日记
公众号：宫水三叶的刷题日记

```java
class Solution {
    public String removeDuplicates(String s) {
        char[] cs = s.toCharArray();
        Deque<Character> d = new ArrayDeque<>();
        for (char c : cs) {
            if (!d.isEmpty() && d.peekLast().equals(c)) {
                d.pollLast();
            } else {
                d.addLast(c);
            }
        }
        StringBuilder sb = new StringBuilder();
        while (!d.isEmpty()) sb.append(d.pollLast());
        sb.reverse();
        return sb.toString();
    }
}
```

- 时间复杂度：$O(n)$
- 空间复杂度：$O(n)$

(数组模拟) 栈解法

宫水三叶の刷题日记
公众号：宫水三叶的刷题日记

```java
class Solution {
    public String removeDuplicates(String s) {
        char[] cs = s.toCharArray();
        char[] d = new char[s.length()];
        int hh = 0, tt = -1;
        for (char c : cs) {
            if (hh <= tt && d[tt] == c) {
                tt--;
            } else {
                d[++tt] = c;
            }
        }
        StringBuilder sb = new StringBuilder();
        while (hh <= tt) sb.append(d[tt--]);
        sb.reverse();
        return sb.toString();
    }
}
```

- 时间复杂度：$O(n)$
- 空间复杂度：$O(n)$

(自带) 双端队列解法

宫水三叶の刷题日记
公众号：宫水三叶的刷题日记

```Java
class Solution {
    public String removeDuplicates(String s) {
        char[] cs = s.toCharArray();
        Deque<Character> d = new ArrayDeque<>();
        for (char c : cs) {
            if (!d.isEmpty() && d.peekLast().equals(c)) {
                d.pollLast();
            } else {
                d.addLast(c);
            }
        }
        StringBuilder sb = new StringBuilder();
        while (!d.isEmpty()) sb.append(d.pollFirst());
        return sb.toString();
    }
}
```

- 时间复杂度：$O(n)$
- 空间复杂度：$O(n)$

(数组模拟) 双端队列解法

宫水三叶の刷题日记
公众号：宫水三叶的刷题日记

```java
class Solution {
    public String removeDuplicates(String s) {
        char[] cs = s.toCharArray();
        char[] d = new char[s.length()];
        int hh = 0, tt = -1;
        for (char c : cs) {
            if (hh <= tt && d[tt] == c) {
                tt--;
            } else {
                d[++tt] = c;
            }
        }
        StringBuilder sb = new StringBuilder();
        while (hh <= tt) sb.append(d[hh++]);
        return sb.toString();
    }
}
```

- 时间复杂度：$O(n)$
- 空间复杂度：$O(n)$

纯数组解法

宫水三叶の刷题日记
公众号：宫水三叶的刷题日记

```java
class Solution {
    public String removeDuplicates(String s) {
        char[] cs = s.toCharArray();
        char[] d = new char[s.length()];
        int hh = 0, tt = -1;
        for (char c : cs) {
            if (hh <= tt && d[tt] == c) {
                tt--;
            } else {
                d[++tt] = c;
            }
        }
        return new String(d, 0, tt + 1);
    }
}
```

- 时间复杂度：$O(n)$
- 空间复杂度：$O(n)$

**🍥 更多精彩内容，欢迎关注：公众号 / Github / LeetCode / 知乎 **

题目描述

这是 LeetCode 上的 **1190. 反转每对括号间的子串** ，难度为 **中等**。

Tag：「双端队列」、「栈」

给出一个字符串 s（仅含有小写英文字母和括号）。

请你按照从括号内到外的顺序，逐层反转每对匹配括号中的字符串，并返回最终的结果。

注意，您的结果中 不应 包含任何括号。

示例 1：

```
输入：s = "(abcd)"
输出："dcba"
```

示例 2：

宫水三叶の刷题日记
公众号：宫水三叶的刷题日记

```
输入：s = "(u(love)i)"
输出："iloveu"
```

示例 3：

```
输入：s = "(ed(et(oc))el)"
输出："leetcode"
```

示例 4：

```
输入：s = "a(bcdefghijkl(mno)p)q"
输出："apmnolkjihgfedcbq"
```

提示：

- 0 <= s.length <= 2000
- s 中只有小写英文字母和括号
- 我们确保所有括号都是成对出现的

基本分析

根据题意，我们可以设计如下处理流程：

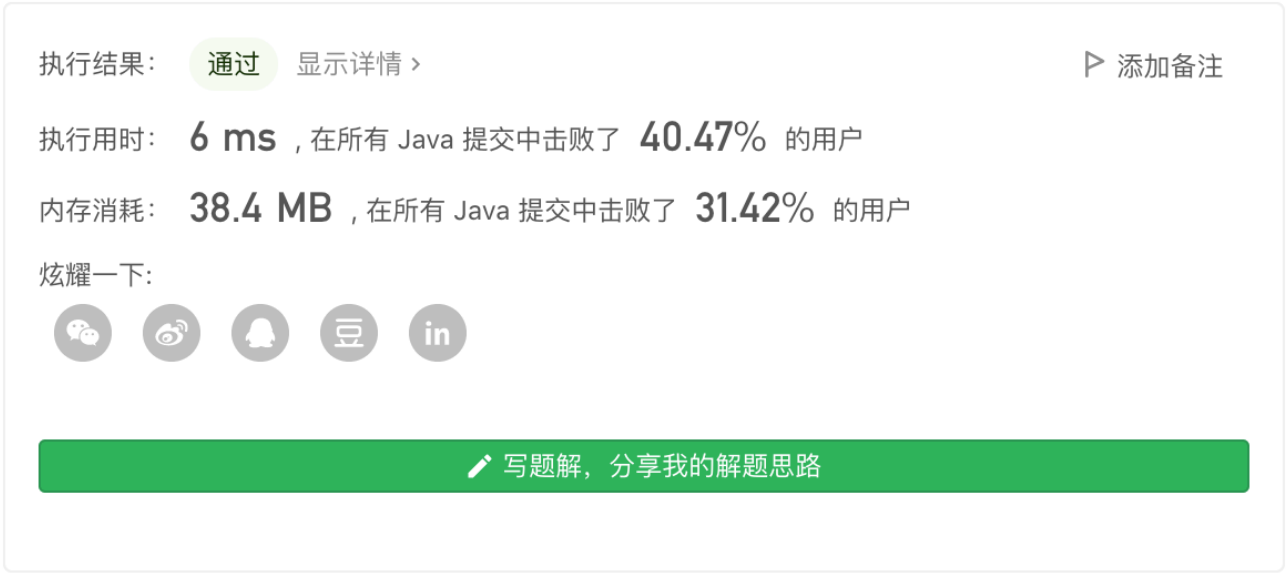
- 从前往后遍历字符串，将不是 `)` 的字符串从「尾部」放入队列中
- 当遇到 `)` 时，从队列「尾部」取出字符串，直到遇到 `(` 为止，并对取出字符串进行翻转
- 将翻转完成后字符串重新从「尾部」放入队列
- 循环上述过程，直到原字符串全部出来完成
- 从队列「头部」开始取字符，得到最终的答案

可以发现，上述过程需要用到双端队列（或者栈，使用栈的话，需要在最后一步对取出字符串再进行一次翻转）。

在 Java 中，双端队列可以使用自带的 `ArrayDeque`，也可以直接使用数组进行模拟。

宫水三叶 刷题日记 公众号：宫水三叶的刷题日记

语言自带双端队列



代码：

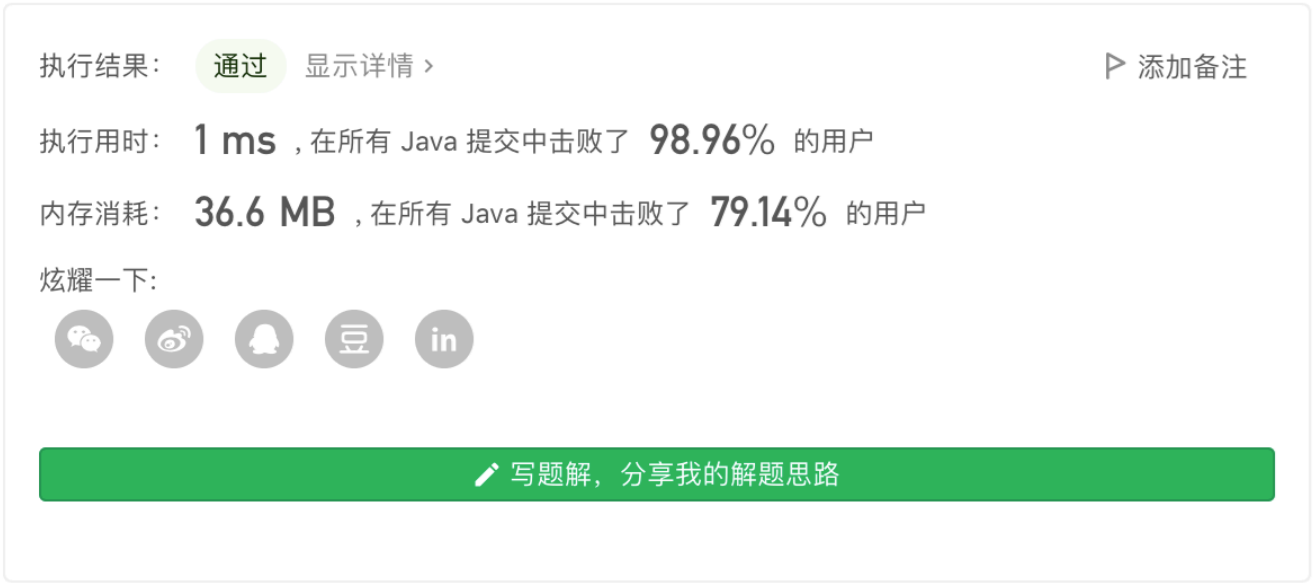
宫水三叶の刷题日记
公众号：宫水三叶的刷题日记

```Java
class Solution {
    public String reverseParentheses(String s) {
        int n = s.length();
        char[] cs = s.toCharArray();
        Deque<Character> d = new ArrayDeque<>();
        for (int i = 0; i < n; i++) {
            char c = cs[i];
            if (c == ')') {
                StringBuilder path = new StringBuilder();
                while (!d.isEmpty()) {
                    if (d.peekLast() != '(') {
                        path.append(d.pollLast());
                    } else {
                        d.pollLast();
                        for (int j = 0; j < path.length(); j++) {
                            d.addLast(path.charAt(j));
                        }
                        break;
                    }
                }
            } else {
                d.addLast(c);
            }
        }
        StringBuilder sb = new StringBuilder();
        while (!d.isEmpty()) sb.append(d.pollFirst());
        return sb.toString();
    }
}
```

- 时间复杂度：每个 `(` 字符只会进出队列一次；`)` 字符串都不会进出队列，也只会被扫描一次；分析的重点在于普通字符，可以发现每个普通字符进出队列的次数取决于其右边的 `)` 的个数，最坏情况下每个字符右边全是右括号，因此复杂度可以当做 $O(n^2)$，但实际计算量必然取不满 n^2，将普通字符的重复弹出均摊到整个字符串处理过程，可以看作是每个字符串都被遍历常数次，复杂度为 $O(n)$
- 空间复杂度：$O(n)$

宫水三叶の刷题日记
公众号：宫水三叶的刷题日记

数组模拟双端队列



代码：

宫水三叶の刷题日记
公众号：宫水三叶的刷题日记

```Java
class Solution {
    char[] deque = new char[2009];
    int head = 0, tail = -1;
    char[] path = new char[2009];
    public String reverseParentheses(String s) {
        int n = s.length();
        char[] cs = s.toCharArray();
        for (int i = 0; i < n; i++) {
            char c = cs[i];
            if (c == ')') {
                int idx = 0;
                while (tail >= head) {
                    if (deque[tail] == '(') {
                        tail--;
                        for (int j = 0; j < idx; j++) {
                            deque[++tail] = path[j];
                        }
                        break;
                    } else {
                        path[idx++] = deque[tail--];
                    }
                }
            } else {
                deque[++tail] = c;
            }
        }
        StringBuilder sb = new StringBuilder();
        while (tail >= head) sb.append(deque[head++]);
        return sb.toString();
    }
}
```

- 时间复杂度：每个 `(` 字符只会进出队列一次；`)` 字符串都不会进出队列，也只会被扫描一次；分析的重点在于普通字符，可以发现每个普通字符进出队列的次数取决于其右边的 `)` 的个数，最坏情况下每个字符右边全是右括号，因此复杂度可以当做 $O(n^2)$，但实际计算量必然取不满 n^2，将普通字符的重复弹出均摊到整个字符串处理过程，可以看作是每个字符串都被遍历常数次，复杂度为 $O(n)$
- 空间复杂度：$O(n)$

**🍥 更多精彩内容，欢迎关注：公众号 / Github / LeetCode / 知乎 **

宫水三叶の刷题日记
公众号: 宫水三叶的刷题日记

💡**更新 Tips：本专题更新时间为 2021-10-07，大概每 2-4 周 集中更新一次。**

最新专题合集资料下载，可关注公众号「宫水三叶的刷题日记」，回台回复「队列」获取下载链接。

觉得专题不错，可以请作者吃糖 🍭🍭🍭 ：

版权声明：任何形式的转载请保留出处 Wiki。