欢迎大家参与本项目，贡献其他语言版本的代码，拥抱开源，让更多学习算法的小伙伴们收益！

> 反转个字符串还有这么多用处？

题目：剑指Offer58-II.左旋转字符串

https://leetcode-cn.com/problems/zuo-xuan-zhuan-zi-fu-chuan-lcof/

字符串的左旋转操作是把字符串前面的若干个字符转移到字符串的尾部。请定义一个函数实现字符串左旋转操作的功能。比如，输入字符串"abcdefg"和数字2，该函数将返回左旋转两位得到的结果"cdefgab"。

示例 1：
输入: s = "abcdefg", k = 2
输出: "cdefgab"

示例 2：
输入: s = "lrloseumgh", k = 6
输出: "umghlrlose"

限制：
1 <= k < s.length <= 10000

思路

为了让本题更有意义，提升一下本题难度：**不能申请额外空间，只能在本串上操作。**

不能使用额外空间的话，模拟在本串操作要实现左旋转字符串的功能还是有点困难的。

那么我们可以想一下上一题目字符串：花式反转还不够！中讲过，使用整体反转+局部反转就可以实现，反转单词顺序的目的。

这道题目也非常类似，依然可以通过局部反转+整体反转 达到左旋转的目的。

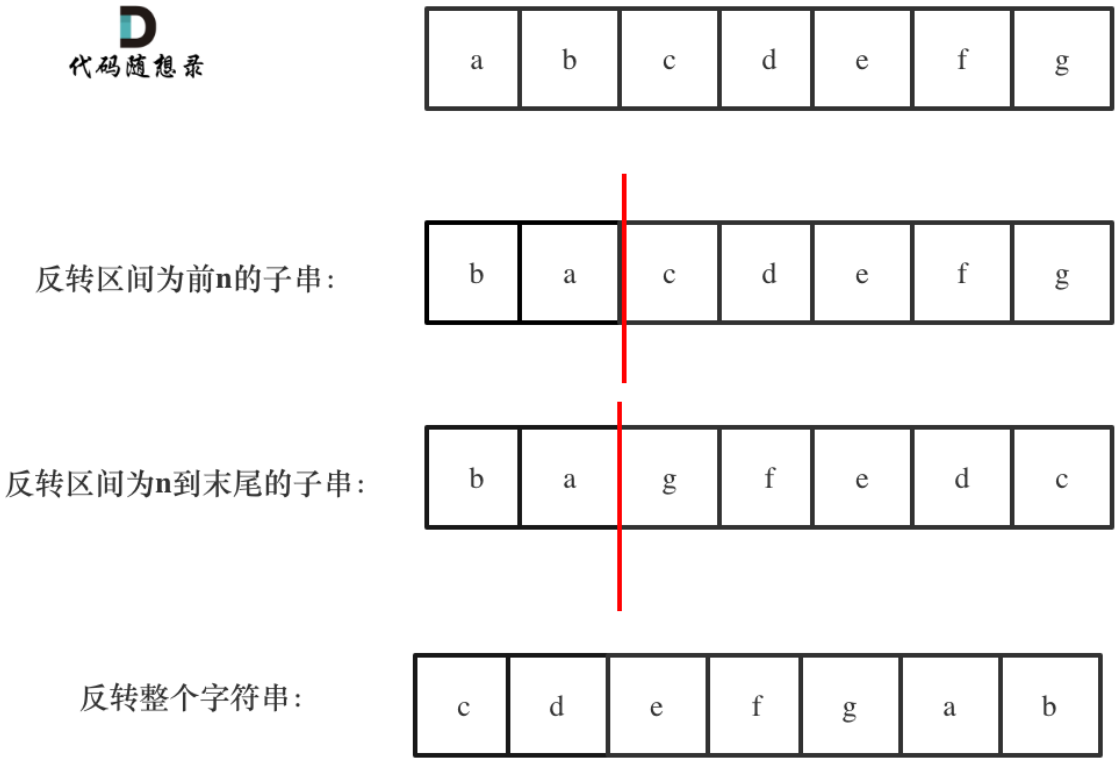
具体步骤为：

1. 反转区间为前n的子串
2. 反转区间为n到末尾的子串
3. 反转整个字符串

最后就可以得到左旋n的目的，而不用定义新的字符串，完全在本串上操作。

例如：示例1中 输入：字符串abcdefg，n=2

如图：



最终得到左旋2个单元的字符串：cdefgab

思路明确之后，那么代码实现就很简单了

C++代码如下：

```
class Solution {
public:
    string reverseLeftWords(string s, int n) {
        reverse(s.begin(), s.begin() + n);
        reverse(s.begin() + n, s.end());
        reverse(s.begin(), s.end());
```

```
        return s;
    }
};
```

是不是发现这代码也太简单了，哈哈。

总结

此时我们已经反转好多次字符串了，来一起回顾一下吧。

在这篇文章344.反转字符串，第一次讲到反转一个字符串应该怎么做，使用了双指针法。

然后发现541. 反转字符串II，这里开始给反转加上了一些条件，当需要固定规律一段一段去处理字符串的时候，要想想在在for循环的表达式上做做文章。

后来在151.翻转字符串里的单词中，要对一句话里的单词顺序进行反转，发现先整体反转再局部反转 是一个很妙的思路。

最后再讲到本题，本题则是先局部反转再 整体反转，与151.翻转字符串里的单词类似，但是也是一种新的思路。

好了，反转字符串一共就介绍到这里，相信大家此时对反转字符串的常见操作已经很了解了。

题外话

一些同学热衷于使用substr，来做这道题。 其实使用substr 和 反转 时间复杂度是一样的 ，都是O(n)，但是使用substr申请了额外空间，所以空间复杂度是O(n)，而反转方法的空间复杂度是O(1)。

如果想让这套题目有意义，就不要申请额外空间。

其他语言版本

Java：

```
class Solution {
    public String reverseLeftWords(String s, int n) {
        int len=s.length();
        StringBuilder sb=new StringBuilder(s);
        reverseString(sb,0,n-1);
        reverseString(sb,n,len-1);
        return sb.reverse().toString();
    }
     public void reverseString(StringBuilder sb, int start, int end) {
        while (start < end) {
            char temp = sb.charAt(start);
            sb.setCharAt(start, sb.charAt(end));
            sb.setCharAt(end, temp);
            start++;
            end--;
            }
```

```
        }
}
```

```python
# 方法一：可以使用切片方法
class Solution:
    def reverseLeftWords(self, s: str, n: int) -> str:
        return s[n:] + s[0:n]

# 方法二：也可以使用上文描述的方法，有些面试中不允许使用切片，那就使用上文作者提到的方法
# class Solution:
#     def reverseLeftWords(self, s: str, n: int) -> str:
#         s = list(s)
#         s[0:n] = list(reversed(s[0:n]))
#         s[n:] = list(reversed(s[n:]))
#         s.reverse()

#         return "".join(s)


# 时间复杂度：O(n)
# 空间复杂度：O(n)，python的string为不可变，需要开辟同样大小的list空间来修改
```

Go：

```go
func reverseLeftWords(s string, n int) string {
    b := []byte(s)
    // 1. 反转前n个字符
    // 2. 反转第n到end字符
    // 3. 反转整个字符
    reverse(b, 0, n-1)
    reverse(b, n, len(b)-1)
    reverse(b, 0, len(b)-1)
    return string(b)
}
// 切片是引用传递
func reverse(b []byte, left, right int){
    for left < right{
        b[left], b[right] = b[right],b[left]
        left++
        right--
    }
}
```

- 作者微信：程序员Carl
- B站视频：代码随想录
- 知识星球：代码随想录