翻转字符串、左旋字符串

# 翻转字符串

## 题目要求：将字符串翻转。

## 利用StringBuilder的reverse方法

利用JDK提供的方法：首先将字符串转换成StringBuilder或StringBuffer，然后调用reverse方法，翻转字符串，最后再转成String即可。

下面查看下：StringBuilder中的reverse实现原理：

@Override

public StringBuilder reverse() {

**super.reverse();**

return this;

}

/\*\*

\*也是交换字符数组前后位置字符实现的

\*/

public AbstractStringBuilder **reverse**() {

boolean hasSurrogates = false;

int n = count - 1;

for (int j = **(n-1) >> 1**; j >= 0; j--) {//取一半

**int k = n - j;//后面对应索引**

**char cj = value[j];**

**char ck = value[k];**

value[j] = ck;

value[k] = cj;

if (Character.isSurrogate(cj) ||

Character.isSurrogate(ck)) {

hasSurrogates = true;

}

}

if (hasSurrogates) {

reverseAllValidSurrogatePairs();

}

return this;

}

## 自己实现的字符串翻转

/\*\*

\* 翻转字符串

\*/

public String **reverseString**(String str){

if(str == null||"".equals(str)) return str;

return reverseString(str,0,str.length()-1);

}

/\*\*

\* 翻转[start,end]范围内的字符串

\*/

public String reverseString(String str,int start,int end){

if(str== null||"".equals(str)) return str;

char[] chars = str.toCharArray();

reverseChars(chars,start,end);

return new String(chars);

}

/\*\*

\* 翻转字符数组的[start,end]部分

\*/

public void reverseChars(char[] chars,int start,int end){

if(start <0 || end > chars.length-1) return;

while(start < end){

char temp = chars[start];

chars[start] = chars[end];

chars[end] = temp;

start++;

end--;

}

}

# 翻转单词顺序

## 题目介绍

**翻转单词顺序**：输入一个英文句子，翻转句子中单词的顺序，但单词内字符的顺序不变。为简单起见，标点符号和普通字母一样处理。

例如输入：We are family.

输出：family. are We

## 思路分析

两次翻转过程即可。第一次翻转：对整个英文句子翻转：.ylimaf era eW

第二次翻转：对句子中的单词逐个翻转：family. are We

因此，需要构造一个翻转方法reverse，两次翻转都通过调用该方法完成即可。

具体思路：将字符串转换成字符数组，然后编写特定段[start,end]的字符翻转reverseChars方法，首先调用该方法实现整个句子的翻转，然后利用遍历方法通过空格判断单词，实现单词翻转。

## Java代码

/\*\*

\* 翻转字符数组的**[start,end]部分**

\*/

public void **reverseChars**(char[] chars,int start,int end){

**if(start <0 || end > chars.length-1) return;**

while(start < end){

char temp = chars[start];

chars[start] = chars[end];

chars[end] = temp;

**start++;**

**end--;**

}

}

/\*\*

\* 翻转单词顺序

\*/

public String **reverseEnglishSentence**(String sentence){

if(sentence==null||"".equals(sentence)) return sentence;

char[] chars = sentence.toCharArray();

**reverseChars(chars,0,chars.length-1);//翻转整个句子**

**//下面逐个翻转单词**

**int wordStart = 0,wordEnd = 0;**

while(wordStart < chars.length){

if(chars[wordStart]==' '&&chars[wordEnd]==' '){//都指向空格

wordStart++;

wordEnd++;

}else if(wordEnd==chars.length||chars[wordEnd]==' '){

**reverseChars(chars,wordStart,wordEnd-1);**

wordStart = wordEnd;

}else{

wordEnd++;

}

}

return new String(chars);

}

# 翻转单词顺序列

## 题目介绍

/\*\*\*\*\*\*剑指Offer58：翻转单词顺序列\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

\* 题目描述

\* 牛客最近来了一个新员工Fish，每天早晨总是会拿着一本英文杂志，写些句子在本子上。同事Cat对Fish写的内容颇感兴趣，有一天他向Fish借来翻看，但却读不懂它的意思。

\* 例如，“student. a am I”。后来才意识到，这家伙原来把句子单词的顺序翻转了，

\* 正确的句子应该是“I am a student.”。Cat对一一的翻转这些单词顺序可不在行，

\* 你能帮助他么？

## 思路分析

\* 思路：首先翻转整个句子，然后再一次翻转每一个单词。

\* 工具方法：翻转start和end范围内的字符。

\* 句子翻转：start=0；end=length-1；

\* 单词翻转：寻找每一个单词是关键。寻找空格，一个空格表示一个单词的结束；

\* 特别注意：最后一个单词是结尾，后面没有空格。

\* 测试： 特殊用例：空字符串；只有一个空格(重要)。

## Java代码

public String **ReverseSentence**(String str) {

**if(str == null || str.length()==0) return "";**

if(str.length() == 1) return str;

char[] chars = str.toCharArray();

int start = 0,end = chars.length-1;

reverseChars(chars,start,end);//翻转整个字符串

int left = 0,right = 0;

while(right < chars.length){//结束标志，右索引小于end

while(left < chars.length && chars[left] == ' ') left++;//排除多个空格

right = left + 1;

while(right < chars.length && chars[right] != ' ') right++;//单词结束了

reverseChars(chars,left,right-1);//翻转该单词

left = ++right;//调整到下一单词开始

}

return new String(chars);

}

//翻转字符数组中start到end的字符段

public void **reverseChars**(char[] chars,int start,int end){

if(start >= end) return;

if(end >= chars.length) return;//防止数组越界

while(start < end){

swap(chars,start++,end--);

}

}

//交换字符数组中两个字符

public void **swap**(char[] chars,int i,int j){

char temp = chars[i];

chars[i] = chars[j];

chars[j] = temp;

}

# 左旋字符串

## 题目介绍

汇编语言中有一种移位指令叫做循环左移（ROL），现在有个简单的任务，就是用字符串模拟这个指令的运算结果。**对于一个给定的字符序列S，请你把其循环左移K位后的序列输出。**

例如，字符序列S=”abcXYZdef”,要求输出循环左移3位后的结果，即“XYZdefabc”。是不是很简单？OK，搞定它！

注意：不能分配额外的内存空间。

## 思路分析

这道题属于思维迁移题目。基于上述的翻转单词顺序，迁移过来解决该问题。

分析：

原字符串： We are family.

整个翻转之后：.ylimaf er**a eW**

左旋4个单位：re family.We a

从上述可以看出，整个翻转之后，将原字符串分成两部分：[0,n-1]和[n,len-1]，只需要把这两部分再分别翻转一次即可。

当然这里顺序可以颠倒，可以先翻转部分，在翻转整体；也可以先整体，再部分。

这里扩展了一下功能：n如果小于零，相当于右转|n|个字符；大于零，左转n个字符；等于零，不变化；|n|若大于字符串长度，取余数。

## Java代码

方法1：先部分翻转后整体翻转。

/\*\*

\* 左旋字符串/右旋字符串

\*/

public String leftRotateString(String str,int n){

if(str == null || "".equals(str)) return str;

char[] chars = str.toCharArray();

n = n%chars.length;

if(n < 0) n += chars.length;

**reverseChars(chars,0,n-1);//前一部分**

**reverseChars(chars,n,chars.length-1);//后一部分**

**reverseChars(chars,0,chars.length-1);//整体翻转**

return new String(chars);

}

/\*\*

\* 翻转字符数组的[start,end]部分(与翻转字符串的方法相同)

\*/

public void **reverseChars**(char[] chars,int start,int end){

if(start <0 || end > chars.length-1) return;

while(start < end){

char temp = chars[start];

chars[start] = chars[end];

chars[end] = temp;

start++;

end--;

}

}

方法2：开辟新的字符数组

/\*\*

\* 另外开辟了一个字符数组，耗费了空间内存，一般不允许

\*/

public String LeftRotateString(String str,int n) {

if(str == null || str.length() == 0) return "";//注意特殊情况

char[] chars = str.toCharArray();

int len = chars.length;

int m = n%len;

if(m<0){m += len;}

**char[] chars1 = new char[len];**

for(int i = 0;i < len;i++){

chars1[i] = chars[(i + m)%len];

}

return new String(chars1);

}

方法3：利用substring方法

/\*\*

\* 利用String的subString方法

\*/

public String LeftRotateString2(String str,int n) {

if(str == null || str.length() == 0) return "";//注意特殊情况

int len = str.length();

n = n%len;

if(n<0){n += len;}

**str = str + str.substring(0,n);**

**return str.substring(n);**

}