# 反转字符串

### 难度等级

简单

### 题目描述

编写一个函数，其作用是将输入的字符串反转过来。输入字符串以字符数组 char[] 的形式给出。

不要给另外的数组分配额外的空间，你必须原地修改输入数组、使用 O(1) 的额外空间解决这一问题。

你可以假设数组中的所有字符都是 ASCII 码表中的可打印字符。

示例 1：

输入：["h","e","l","l","o"]

输出：["o","l","l","e","h"]

示例 2：

输入：["H","a","n","n","a","h"]

输出：["h","a","n","n","a","H"]

### 迭代算法

1. 设定2个指针start和end，一个指向队首一个指向队尾，交换start和end值的位置
2. 交换后，将2个指针均向队列的中间挪动(即：i+1，j-1)，每挪动一次，交换一次值
3. 当2个指针指向同一个元素或者2个指针的前后顺序改变（end指针的index小于start指针的index）终止继续交换

### 迭代代码

|  |
| --- |
| def reverseString(s):  """  :type s: List[str]  :rtype: None Do not return anything, modify s in-place instead.  """  i = 0#队首  j = len(s)-1#队尾  while i<=j:  s[i],s[j]=s[j],s[i] #交换队首与队尾的位置  i+=1#改变i指针  j-=1#改变j指针  return s |

### 进阶算法

尝试将上面的迭代算法转换为递归的过程后得出递归的三要素：

**递归的结束条件：**start指针的位置大于或end指针

**本层递归做什么：**交换start和end的值的位置，并且继续递归调用函数处理下一个start和end的值

**本层递归返回值：**本层递归原地修改s的元素，而不进行返回

### 进阶代码

|  |
| --- |
| def reverseString(s):  """  Do not return anything, modify s in-place instead.  """  start = 0#队首  end = len(s)-1#队尾  def reverse(s,start,end):  if start>=end:#start与end相遇的时候停止递归  return  s[start],s[end] = s[end],s[start]#交换位置  reverse(s,start+1,end-1)#递归处理下一对start和end  reverse(s,start,end) |