1、给你一个字符串 s，找到 s 中最长的回文子串。

2、字符串反转：

使用 reversed()

```
str = 'Runoob'
print(''.join(reversed(str)))
```

使用字符串切片

```
str = 'Runoob'
print(str[::-1])
```

逆序遍历索引

```
chars = list(s)
for i in range(len(s) // 2):
    tmp = chars[i]
    chars[i] = chars[len(s) - i - 1]
    chars[len(s) - i - 1] = tmp
```

public static String reverseBybuffer(String input) {
String result=new StringBuffer(input).reverse().toString();
return result;
}

3、#最长前缀

def test_longestPrefix(str) -> str:

if not str:
return ''

if len(str)==1:

```
return str[0]
s1=str[0]
for i in range(1,len(str)):
while not str[i].startswith(s1): #print("aaa".startswith("")) 字符串都以“”开头 #print("aaa"[0:0].startswith(""))
字符串没有取到指就是空格
s1=s1[0:len(s1)-1]

return s1


def test_longestPrefix2(str: list):
if not str:
return ''
if len(str)==1:
return str[0]
str.sort() #排序
a=str[0]
b=str[len(list)-1]
result=''
for i in range(len(min)):
if i >= len(b) or a[i] != b[i]:
break
result+=a[i]

return result
```

4、字符串中大小写字母分成前后两部分，字母顺序不变

```
def test_reverseWords(self, s):
"""
:type s: str
:rtype: str
"""
if not s:
return s
data_list = s.split()
length = len(data_list)
```

```
for i in range(0, len(data_list)):
data_list[i], data_list[length - i - 1] = data_list[length - i - 1], data_list[i]

return " ".join(data_list)
```

5、给定一个字符串（长度<8)，按字典序输出所有可能排列

6、有效的括号：（https://leetcode.cn/problems/valid-parentheses/solution/you-xiao-de-gua-hao-by-leetcode-solution/）

给定一个只包括

'(' ,

')' ,

'{' ,

'}' ,

'[' ,

']' 的字符串

s ，判断字符串是否有效。

python：

```
class Solution:
def isValid(self, s: str) -> bool:
if len(s) % 2 == 1:
return False
pairs = {
")": "(",
"]": "[",
"}": "{",
}
stack = list()
for ch in s:
if ch in pairs:
if not stack or stack[-1] != pairs[ch]:
return False
stack.pop()
else:
stack.append(ch)
```

return not stack

Java：

```
class Solution {
public boolean isValid(String s) {
int n = s.length();
if (n % 2 == 1) {
return false;
}

Map<Character, Character> pairs = new HashMap<Character, Character>() {{
put(')', '(');
put(']', '[');
put('}', '{');
}};
Deque<Character> stack = new LinkedList<Character>();
for (int i = 0; i < n; i++) {
char ch = s.charAt(i);
if (pairs.containsKey(ch)) {
if (stack.isEmpty() || stack.peek() != pairs.get(ch)) {
return false;
}
stack.pop();
} else {
stack.push(ch);
}
}
return stack.isEmpty();
}
}
```

7、最长回文子串：（https://leetcode-cn.com/problems/longest-palindromic-substring/solution/zui-chang-hui-wen-zi-chuan-by-leetcode-solution/）

8、数字 n 代表生成括号的对数，请你设计一个函数，用于能够生成所有可能的并且 **有效的** 括号组合。

9、 字符串消消乐，将字符串中相邻相同的字符一起消掉，最后输出消除完成的字符串

示例：abcccbxezzzrf7788fn
输出：axern
说明：从左住右消除，第一趟消除相邻相同的“ccc”、“zzz”、“77”、“88”，
得到abbxerffn，第二趟消除相邻相同的“bb”、“ff”，得到axern，
不存在相邻相同字符，消除结束。

```
def test_getString(a):
s = []
# 前一个被消除的元素
del_str = ''
for i in a:
# 栈为空，直接添加入栈
if len(s) == 0:
s.append(i)
else:
# 判断i 与被前一个被消除的元素是都相等
if i == del_str:
# 如果相等不做处理
continue
# 判断 i 与栈顶元素是否相等
elif i == s[-1]:
# 弹出栈顶元素
del_str = s.pop(-1)
else:
# 入栈
s.append(i)
print( ''.join(s))
```

10、给你一个字符串 date ，按 YYYY-MM-DD 格式表示一个 现行公元纪年法 日期。返回该日期是当年的第几天

https://leetcode.cn/problems/day-of-the-year/solution/

```
class Solution(object):
    def dayOfYear(self, date):
        """
        :type date: str
        :rtype: int
        """
```

```
        year = int(date[0:4])
        month =int(date[5:7])
        day = int(date[8:10])
        sums=0
        nums=[31,28,31,30,31,30,31,30,31,31,30,31]
        if year % 400 ==0 or (year %4 ==0 and year % 100 !=0):
            nums[1]+=1
        for i in range(0,month-1):
            sums =sums+nums[i]
        sums =sums+day
        return sums
```

11、给你一个整数数组 nums 。如果任一值在数组中出现 **至少两次** ，返回 true ；如果数组中每个元素互不相同，返回 false 。

方法1：

```
class Solution(object):
    def containsDuplicate(self, nums):
        """
        :type nums: List[int]
        :rtype: bool
        """
        if not nums:
            return False
        nums.sort()
        for i in range(len(nums)-1):
            if nums[i] ==nums[i+1]:
                return True
        return False
```

方法二：

```
class Solution(object):
    def containsDuplicate(self, nums):
        """
        :type nums: List[int]
        :rtype: bool
        """
```

```
    if not nums:
        return False
    return len(set(nums)) != len(nums)
```

12、对已排序的数组中，找出是否存在相同的元素

输入描述：1，3，5，9

2，6，8

输出描述：False

```
def test2(arr1,arr2):
    if not arr1 or not arr2:
        return False
    i,j=0,0
    while  i <=(len(arr1) -1) and j <= (len(arr2)-1) :
        if arr1[i] ==arr2[j]:
            return  True
        if arr1[i] <arr2[j]:
            i+=1
        else:
            j+=1
    return False
```

13、输入一个字符串，打印出该字符串中字符的所有排列。（https://leetcode.cn/problems/zi-fu-chuan-de-pai-lie-lcof/ ）

class Solution:

def permutation(self, s: str) -> List[str]:

c, res = list(s), []

def dfs(x):

if x == len(c) - 1:

res.append(''.join(c)) # 添加排列方案

return

dic = set()

for i in range(x, len(c)):

if c[i] in dic: continue # 重复，因此剪枝

dic.add(c[i])

c[i], c[x] = c[x], c[i] # 交换，将 c[i] 固定在第 x 位

dfs(x + 1) # 开启固定第 x + 1 位字符

c[i], c[x] = c[x], c[i] # 恢复交换

dfs(0)

return res

作者：jyd

链接：https://leetcode.cn/problems/zi-fu-chuan-de-pai-lie-lcof/solution/mian-shi-ti-38-zi-fu-chuan-de-pai-lie-hui-su-fa-by/

来源：力扣（LeetCode）