# 038 Count and Say

2018年3月29日 17:14

## Question:

The count-and-say sequence is the sequence of integers with the first five terms as following:

1. 1
2. 11
3. 21
4. 1211
5. 111221

1 is read off as "one 1" or 11.

11 is read off as "two 1s" or 21.

21 is read off as "one 2, then one 1" or 1211.

Given an integer *n*, generate the *n*th term of the count-and-say sequence.

Note: Each term of the sequence of integers will be represented as a string.

**Example 1:**

**Input:** 1

**Output:** "1"

**Example 2:**

**Input:** 4

**Output:** "1211"

来自 <https://leetcode.com/problems/count-and-say/description/>

数数并说序列是一个整数序列，第二项起每一项的值为对前一项的计数，其前五项如下：

1. 1
2. 11
3. 21
4. 1211
5. 111221

1 被读作 "一个一" 即 11。

11 被读作 "两个一" 即 21。

21 被读作 "一个二 和 一个一" 即 1211。

给一个正整数 *n* ，输出数数并说序列的第 *n* 项。

注意：该整数序列的每项都输出为字符串。

**例 1:**

**输入:** 1

**输出:** "1"

**例 2:**

**输入:** 4

**输出:** "1211"

来自 <https://leetcode-cn.com/problems/count-and-say/description/>

## Solution for Python3:

```python
class Solution:
    def countAndSay(self, n):
        """
        :type n: int
        :rtype: str
        """
        if n == 1:
            return '1'
        if n == 2:
            return '11'

        result = self.countAndSay(n - 1)
        newresult = ''
        count = 1
        for i in range(1,len(result)):
            if result[i] != result[i - 1]:
                newresult += str(count) + result[i - 1]
                count = 1
            else:
                count += 1
            if i == len(result) - 1:
                newresult += str(count) + result[i]
        return newresult
```

## Solution for C++:

```cpp
class Solution {
public:
    string countAndSay(int n) {
        if (n == 1) {
            return "1";
        }
        if (n == 2) {
            return "11";
        }
        string result = countAndSay(n - 1);
        string newresult = "";
        int count = 1;
        for (int i = 1; i < result.size(); i++) {
            if (result[i] != result[i - 1]) {
                newresult.push_back('0' + count);
                newresult.push_back(result[i - 1]);
                count = 1;
            } else {
                count++;
            }
            if (i == result.size() - 1) {
                newresult.push_back('0' + count);
                newresult.push_back(result[i]);
            }
        }
        return newresult;
    }
};
```

## Appendix:

**Python 类内函数递归调用self.function()**

**同类型才能使用‘+’**