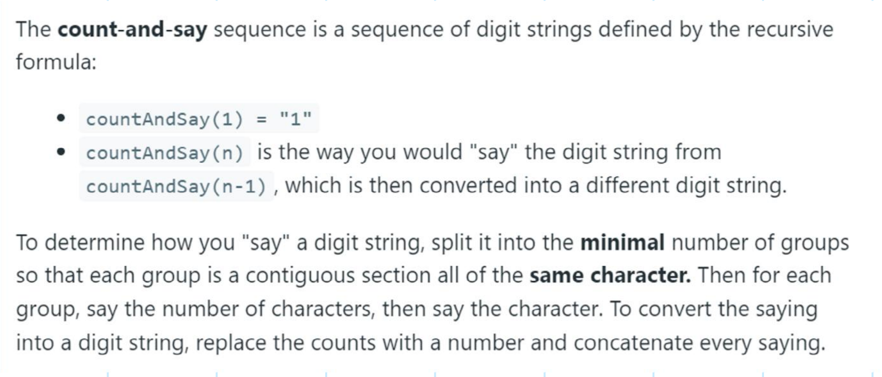
38. Count and Say

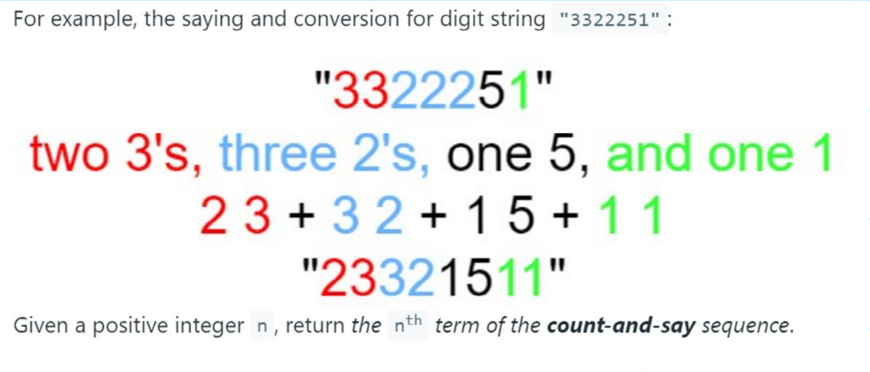
13 February 2022 04:09 PM



The **count-and-say** sequence is a sequence of digit strings defined by the recursive formula:

- `countAndSay(1) = "1"`
- `countAndSay(n)` is the way you would "say" the digit string from `countAndSay(n-1)`, which is then converted into a different digit string.

To determine how you "say" a digit string, split it into the **minimal** number of groups so that each group is a contiguous section all of the **same character.** Then for each group, say the number of characters, then say the character. To convert the saying into a digit string, replace the counts with a number and concatenate every saying.



For example, the saying and conversion for digit string `"3322251"`:

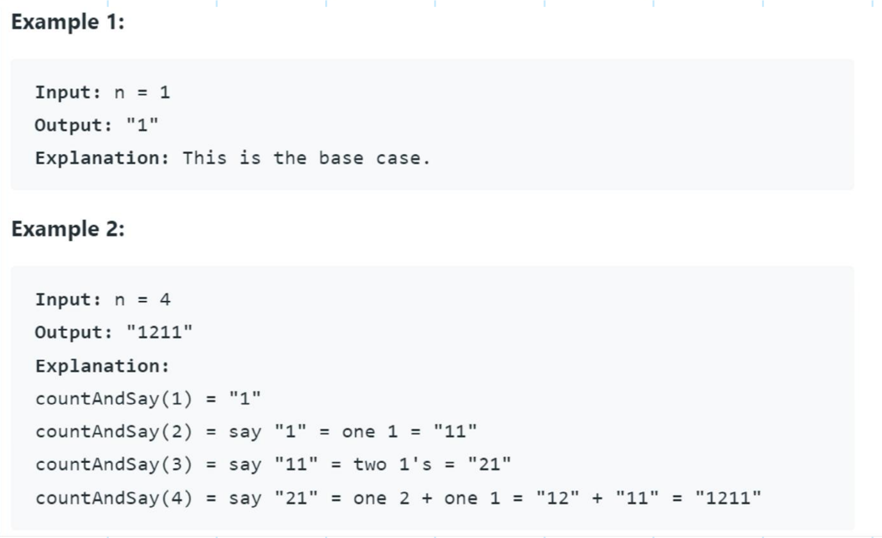
"3322251"

two 3's, three 2's, one 5, and one 1

2 3 + 3 2 + 1 5 + 1 1

"23321511"

Given a positive integer `n`, return *the* `n`th *term of the* ***count-and-say*** *sequence.*



Example 1:

```
Input: n = 1
Output: "1"
Explanation: This is the base case.
```

Example 2:

```
Input: n = 4
Output: "1211"
Explanation:
countAndSay(1) = "1"
countAndSay(2) = say "1" = one 1 = "11"
countAndSay(3) = say "11" = two 1's = "21"
countAndSay(4) = say "21" = one 2 + one 1 = "12" + "11" = "1211"
```

Say(1) = "1"

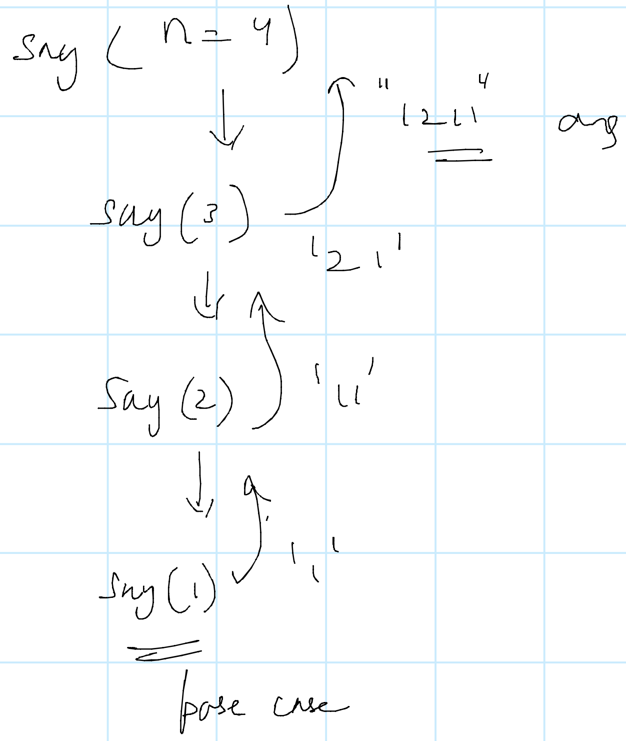
say(2) = "1" = one 1 = "11"

say(3) = "11" = two 1 = "21"

say(4) = "21" = one 2 one 1 = "1211"

for finding 4 you need 3 same for 3 you need 2 so on.

So <u>recursion</u>.



So now if you watch clearly 1st is Frequency 2nd is Element

(F E)

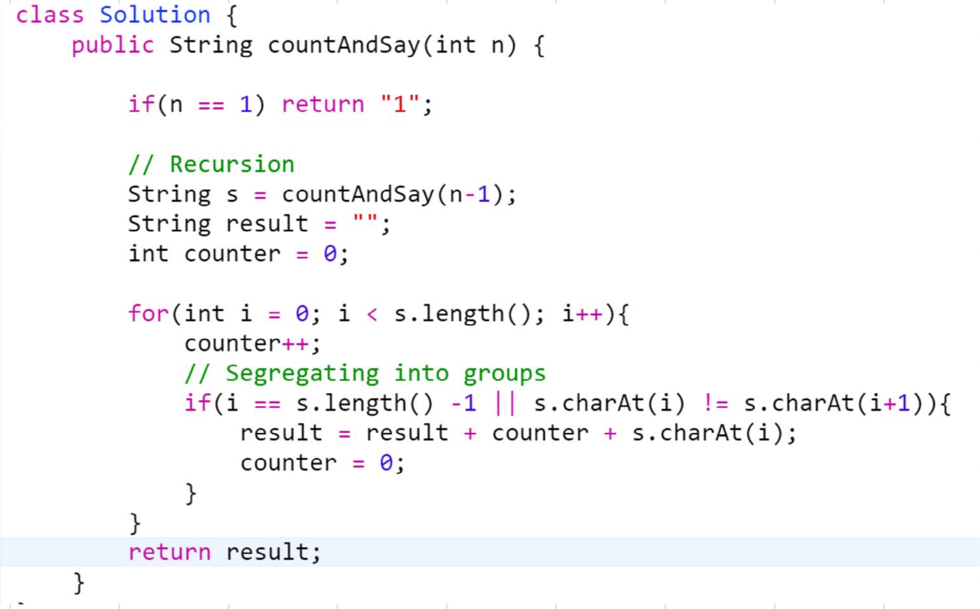
"1 1 1 | 2 2 | 3 | 2 | 1" → "3 1 2 2 1 3 2 1 1"

3 1 22 31 12 11

// Making String

```
counter = 0,  result = 0

for (i = 0, i < d, i++)

        for index out of bound
```

```
f (  ,  .  -
{ counter ++
                         for index out of bound
  if(i==l-1 || s.charAt(i) != s.charAt(i+1))
  {
      rlt = result + counter + s. rAt(i

      coun = 0. // reset the counter

  }
```



```
class Solution {
    public String countAndSay(int n) {

        if(n == 1) return "1";

        // Recursion
        String s = countAndSay(n-1);
        String result = "";
        int counter = 0;

        for(int i = 0; i < s.length(); i++){
            counter++;
            // Segregating into groups
            if(i == s.length() -1 || s.charAt(i) != s.charAt(i+1)){
                result = result + counter + s.charAt(i);
                counter = 0;
            }
        }
        return result;
    }
```