38. Count and Say

Recursive formula:

base: countAndSay(1) = 1

Example:

n = 4

CAS(1) = "1"

CAS(2) = one 1 = "11"

CAS(3) = two 1 = "21"

CAS(4) = one 2 one 1 = "1211"

Analyse:

count(n) is based on count(n-1)

∴ we need $\boxed{\text{count(n-1)}}$ for this recursion

what should we do?

Example: n = 2

CAS(2) = ?? wait to find.

go

back

CAS(1) = "1"

with CAS(1) = "1"

Now, we can start to construct our new string.

"1" -> one 1 -> "1"

Okay.

Now, how to count the string?

Obviously, $\boxed{\text{two\_pointers}}$ would be fantastic

example string a = "333221"

Algorithm:

(1) if a[i] == a[j]:

i stay, j move

a = "333221"

else

adding the counted part to the res string. and move i and j to the same place

a = "333221"

special consideration:

if a = "333222"

process "333222"

↓

"333222"

↓

"333222"

↓

i cannot reach end, but j seems to be okay for every situation.

Then, I guess we should loop j to the end ratter than loop i

/ .

pseudo code:

```
def CAS(n):
    if n==1 :
        return "1"
    else:
        a = CAS(n-1)
        i= 0
        j= 0
        ans= "  "
        while j < len(last):
            if a[i] == a[j]:
                # special situation when j is last
if "333222" (i, j)    ans += str(j-i+1) + a[i]
if "333222!" (i, j)   ans += "1" + a[j]
                j++
            else:
                ans += str(j-i) + a[i
                i= j
```

Above code actually works for the Leetcode submission. However, it

Seems like there's a better way to do the two pointers.

Better two-pointers:

```
while i < len:
    while j < len and a[i] == a[j]
        j++
    ans += str(j-i) + s[i]
    i = j-1
    i += 1
```