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"

题目

就是数上一个字符串中的数字。

例如上一个字符串为1211

下一个就是 1个 1🡪11+

1个 2🡪+12+

1个 1🡪

2个 1🡪+21

结果就饿是111221

这道题典型的采用递归的方法，在前一个元素等于后一个元素且不为最后一个元素的时候计数器++，别的情况都添加，先添加计数器再添加元素。因为这里是逐步生成的string，所以采用stringbuffer

代码

public static String countAndSay(int n,String preSay){

StringBuffer s = new StringBuffer();//因为每次生成的preSay字符串都是逐步生成的，所以采用stringbuffer

// System.out.println(preSay);

int count=1;//计数器，从1开始

for(int i=0;i<preSay.length();i++){

//当前位置的char不为最后一个char时且和下一个位置的char相等的时候计数器++

if(i+1<preSay.length()&&preSay.charAt(i)==preSay.charAt(i+1)){

count++;

}

else {//其他情况就直接输出

s.append(count);//在结果上先添加计数器

s.append(preSay.charAt(i));//然后添加该位置的char

count=1;//计数器回归，准备下一次的计数

}

}

if(n==2)//因为第i个字符串由i-1次递归产生，而且第一次为1直接给的，所以n=2的时候为起点。

return s.toString();

else

return countAndSay(n-1,s.toString());//递归

}