

## Experiment 2.3

Student Name: Yash Gupta  
Branch: BE-CSE  
Semester: 6  
Subject Name: CC LAB

UID: 20BCS5009  
Section/Group: 20BCS\_DM-716 B  
Date of Performance: 27/02/23  
Subject Code: 20CSP\_351

### 1. Aim:

To implement the concept of divide and conquer.

### 2. Objective:

- The objective is to build problem solving capability and to learn the basic concepts of data structures.
- Understand the problem and find out better approach to solve particular problem

### 3. LeetCode code and output:

- Count and Say

The screenshot shows the LeetCode interface for problem 38, "Count and Say". The problem is categorized as "Medium" and has 3.1K likes and 6.6K views. The description explains that the "count-and-say" sequence is defined by a recursive formula:  $\text{countAndSay}(1) = "1"$  and  $\text{countAndSay}(n)$  is the way you would "say" the digit string from  $\text{countAndSay}(n-1)$ , which is then converted into a different digit string. To determine how you "say" a digit string, split it into the minimal number of substrings such that each substring contains exactly one unique digit. Then for each substring, say the number of digits, then say the digit. Finally, concatenate every said digit. 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  $n^{\text{th}}$  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:  
 $\text{countAndSay}(1) = "1"$   
 $\text{countAndSay}(2) = \text{say } "1" = \text{one } 1 = "11"$   
 $\text{countAndSay}(3) = \text{say } "11" = \text{two } 1\text{'s} = "21"$   
 $\text{countAndSay}(4) = \text{say } "21" = \text{one } 2 + \text{one } 1 = "1211"$

class Solution:

```
def countAndSay(self, n: int) -> str:
    dp = [ "" for i in range(0,n+1)]
    dp[1] = "1 "
    i = 2 ;
    while( i < n+1 ):
        print(f'i: {i} dp[{i-1}]: {dp[i-1]}')
        c = 0
        for j in range(0,len(dp[i-1])-1):
            if( dp[i-1][j] == dp[i-1][j+1]):
                c += 1;
            else:
                dp[i] += chr(c+1+ord('0')) + dp[i-1][j] ;
                c=0;
        dp[i] += ' ';
        i +=1
    return dp[-1][:-1];
```

**Yash\_Gupta202**  
 Apr 24, 2023 15:32

Details
 + Solution

Python3

Runtime **54 ms**

Beats **40.32%**

Memory **13.8 MB**

Beats **98.49%**

Click the distribution chart to view more details

Notes

Write your notes here

Related Tags

Select tags 0/5

```

class Solution:
    def countAndSay(self, n: int) -> str:
        dp = [ "" for i in range(0,n+1)]
        dp[1] = "1 "
        i = 2 ;
        while( i < n+1 ):
            print(f'i:{i} dp[{i-1}]:{dp[i-1]}')
            c = 0
            for j in range(0,len(dp[i-1])-1):
                if dp[i-1][j] == dp[i-1][j+1]):
                    c += 1;
                else:
                    dp[i] += chr(c+1+ord('0')) + dp[i-1][j] ;
                    c=0;
            dp[i] += ' ';
            i +=1

        return dp[-1][:-1];

```

Console ^

Run
 Submit

## • Jewels and Stones

[Description](#) [Editorial](#) [Solutions \(5.7K\)](#) [Submissions](#)

### 771. Jewels and Stones

Hint

Easy 4.5K 542

Companies

You're given strings `jewels` representing the types of stones that are jewels, and `stones` representing the stones you have. Each character in `stones` is a type of stone you have. You want to know how many of the stones you have are also jewels.

Letters are case sensitive, so `"a"` is considered a different type of stone from `"A"`.

**Example 1:**

**Input:** `jewels = "aA", stones = "aAAbbbb"`  
**Output:** 3

**Example 2:**

**Input:** `jewels = "z", stones = "ZZ"`  
**Output:** 0

**Constraints:**

- `1 <= jewels.length, stones.length <= 50`
- `jewels` and `stones` consist of only English letters.
- All the characters of `jewels` are **unique**.

Accepted **875.3K** | Submissions **992.2K** | Acceptance Rate **88.2%**

Seen this question in a real interview before? 1/4

Yes No

Discussion (23)

class Solution:

```
def numJewelsInStones(self, jewels: str, stones: str) -> int:
```

```
    dictj={ }
```

```
    for i in jewels:
```

```
        if i in dictj:
```

```
            dictj[i]+=1
```

```
        else:
```

```
            dictj[i]=1
```


```
    sum=0
```

```
    for j in stones:
```

```
        if j in dictj:
```

```
            sum+=dictj[j]
```

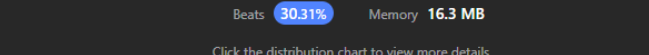
```
    return(sum)
```



**Yash\_Gupta202**  
 May 13, 2023 19:47

[Details](#)
[+ Solution](#)

Python3



Runtime **39 ms**      Beats **30.31%**      Memory **16.3 MB**      Beats **12.23%**

[Click the distribution chart to view more details](#)

**Notes**

**Related Tags**

0/5

```

class Solution:
    def numJewelsInStones(self, jewels: str, stones: str) -> int:
        dictj={}
        for i in jewels:
            if i in dictj:
                dictj[i]+=1
            else:
                dictj[i]=1
        sum=0
        for j in stones:
            if j in dictj:
                sum+=dictj[j]

        return(sum)
          
```

**Console** ^

⌵
Run
Submit