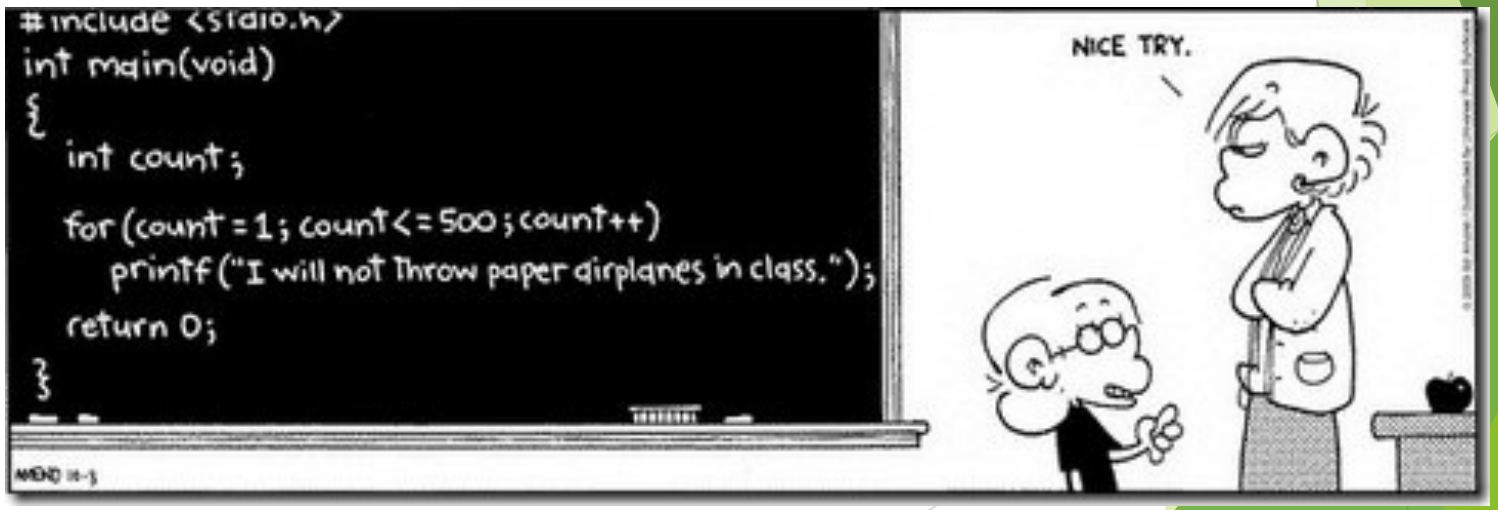
String Manipulation

Algorithm Fundamentals - Hosted by WiCSE



Agenda

10 min - Review the **warm-up problem**

30 min – Independent work on **main problem**

- Optional: Work on the bonus problem if you finish early!

10 min – Review the **main problem** solution

10 min – Wrap-up

Goals

- Create a safe space to practice and build confidence around algorithm fundamentals and data structures
- Actively engage in the discussion of potential solutions to the problems
- Foster a community of study buddies. Join our Teams Channel!
- Have Fun!

String Manipulation Definition

"The action of the fundamental operations on strings, including their **creation**, **concatenation**, the **extraction of string segments**, **string matching**, their **comparison**, **discovering their length**, **replacing substrings** by other strings, **storage**, and **input/output**."

- encyclopedia.com/computing

Warm Up:
Valid Palindrome

125. Valid Palindrome

Easy 2524 4412 Add to List Share

Given a string `s`, determine if it is a palindrome, considering only alphanumeric characters and ignoring cases.

Example 1:

```
Input: s = "A man, a plan, a canal: Panama"
Output: true
Explanation: "amanaplanacanalpanama" is a palindrome.
```

Example 2:

```
Input: s = "race a car"
Output: false
Explanation: "raceacar" is not a palindrome.
```

Constraints:

- `1 <= s.length <= 2 * 10^5`
- `s` consists only of printable ASCII characters.

Main Problem: *Count and Say*

38. Count and Say

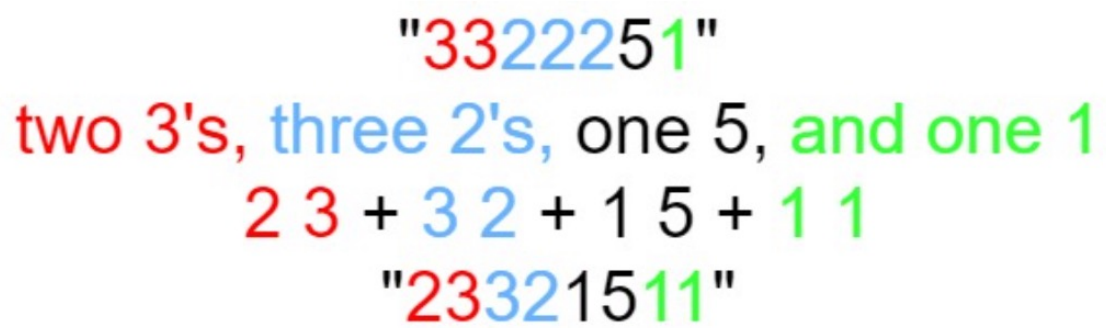
Medium 893 2571 Add to List Share

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"`:



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"
```

Constraints:

- `1 <= n <= 30`

Independent Hack Time!

- **Warm Up:** Valid Palindrome
  - https://leetcode.com/problems/valid-palindrome/
- **Main:** Count and Say
  - https://leetcode.com/problems/count-and-say/
- **Optional Bonus:** Longest Palindromic Substring
  - https://leetcode.com/problems/longest-palindromic-substring/
- **Next Month's Warm Up (Arrays):** Best Time to Buy And Sell Stock
  - https://leetcode.com/problems/best-time-to-buy-and-sell-stock/

Main Problem: *Count and Say*

38. Count and Say

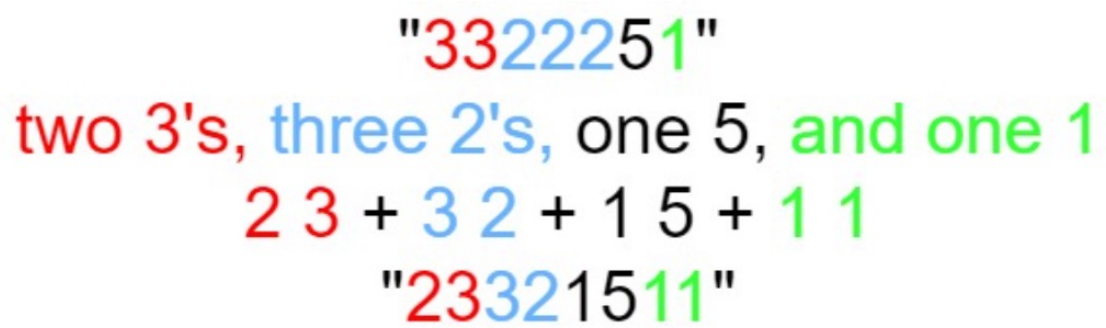
Medium 893 2571 Add to List Share

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"`:



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"
```

Constraints:

- `1 <= n <= 30`

Optional Bonus Problem: *Longest Palindromic Substring*

Constraints:

- `1 <= s.length <= 1000`
- `s` consist of only digits and English letters.

5. Longest Palindromic Substring

Medium 13830 817 Add to List Share

Given a string `s`, return *the longest palindromic substring* in `s`.

Example 1:

```
Input: s = "babad"
Output: "bab"
Note: "aba" is also a valid answer.
```

Example 2:

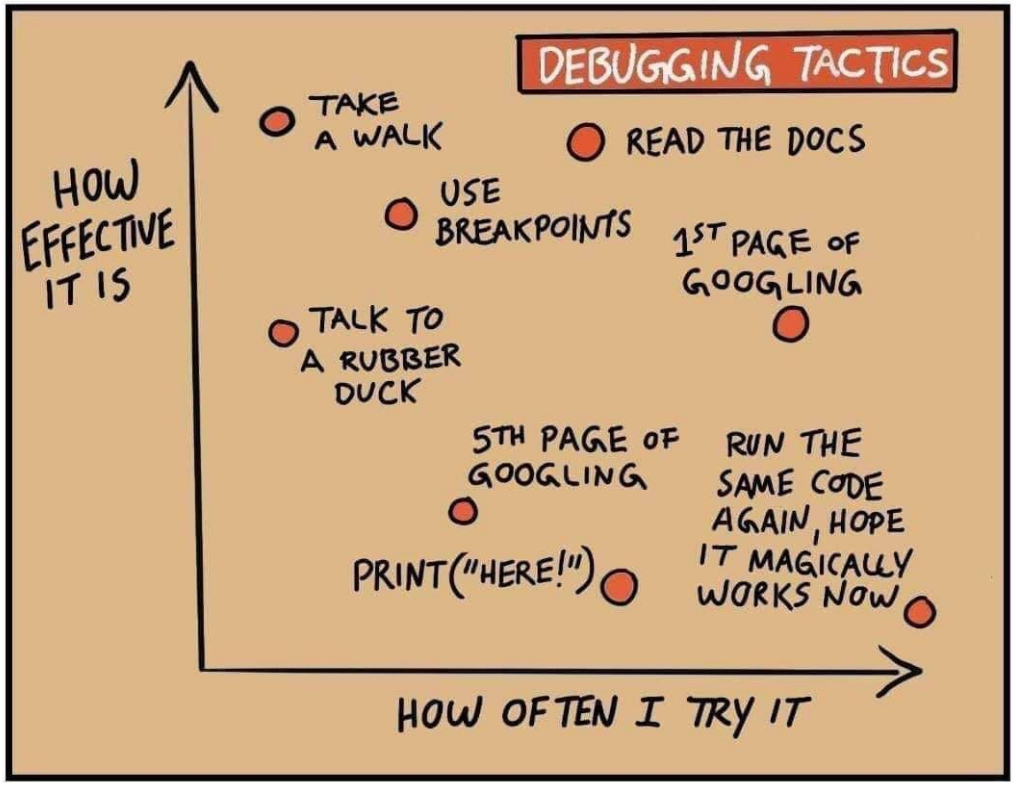
```
Input: s = "cbbd"
Output: "bb"
```

Example 3:

```
Input: s = "a"
Output: "a"
```

Example 4:

```
Input: s = "ac"
Output: "a"
```



Thank you! See you next month for **Arrays**!