Programming Practice in String

🔹 Easy (1–10)

These questions test basic string manipulation and built-in functions.

1. **Count Vowels and Consonants**
   Write a program to count the number of vowels and consonants in a given string.
2. **Reverse a String**
   Reverse the input string without using any built-in reverse functions.
3. **Check Palindrome**
   Check whether a given string is a palindrome (same forwards and backwards).
4. **Remove Punctuation from String**
   Remove all punctuation characters from a string.
5. **Replace All Spaces with Hyphen**
   Replace all spaces in a sentence with -.
6. **Count Word Frequency**
   Given a string, count the frequency of each word and print the result as a dictionary.
7. **Check Anagram**
   Check if two input strings are anagrams (contain the same characters with same frequencies).
8. **Capitalize First Letter of Each Word**
   Write a function to capitalize the first letter of each word in a sentence.
9. **Find the Longest Word**
   Given a string of words, return the longest word.
10. **Character Frequency Count**
    Print the frequency of each character in a string (excluding spaces).

🔸 Medium (11–16)

These require slightly deeper string manipulation, conditions, or loops.

11. **Compress a String**
    Compress the string using counts of repeated characters. For example:
    Input: "aaabbcccc" → Output: "a3b2c4"
12. **Group Words with Same Set of Characters**
    Input: List of strings → Group all words that are made up of the same set of characters (e.g., ["act", "cat", "tac", "tap", "pat"] → [["act", "cat", "tac"], ["tap", "pat"]])
13. **Extract Digits from a String**
    Write a program to extract all digits from a given string and return them as a list of integers.

14. **Check for Substring Presence Without in Operator**
    Manually check if one string is a substring of another without using in or .find().

15. **Validate Password Strength**
    Check whether a password is valid based on these rules:
    - At least 8 characters
    - Contains uppercase, lowercase, digit, and special character

16. **Convert Roman Numeral to Integer**
    Input a string containing a Roman numeral and convert it to an integer.

🔻 **Hard (17–20)**

These require algorithmic thinking and multi-step string operations.

17. **Zigzag Conversion of a String**
    Write a function that takes a string and an integer numRows, and returns the string written in a zigzag pattern (like in leetcode's zigzag conversion problem).

18. **Find All Permutations of a String**
    Generate all possible permutations of a given string without using itertools.

19. **Longest Substring Without Repeating Characters**
    Write a program that finds the length of the longest substring without repeating characters.
    E.g., Input: "abcabcbb" → Output: 3 (for "abc")

20. **Word Pattern Matching**
    Given a pattern and a string, check if the string follows the same word pattern.
    E.g., pattern = "abba", string = "dog cat cat dog" → True