Basic Level Questions

1. Palindrome Check

Problem Statement:

Write a Java program to check if a given string is a palindrome. A palindrome is a word that reads the same forward and backward.

Test Cases:

- Input: `"madam"` → Output: `true`
- Input: `"hello"` → Output: `false`
- Input: `"racecar"` → Output: `true`
- Input: `"Noon"` → Output: `true` (Ignore case)

---

2. Count Vowels & Consonants

Problem Statement:

Write a program to count the number of vowels and consonants in a given string.

Test Cases:

- Input: `"hello"` → Output: `Vowels: 2, Consonants: 3`
- Input: `"Java"` → Output: `Vowels: 2, Consonants: 2`
- Input: `"aeiou"` → Output: `Vowels: 5, Consonants: 0`
- Input: `"BCDF"` → Output: `Vowels: 0, Consonants: 4`

---

3. Reverse a String

Problem Statement:

Write a program to reverse a given string using `StringBuilder`.

Test Cases:

- Input: `"hello"` → Output: `"olleh"`

- Input: `"Java"` → Output: `"avaJ"`
- Input: `"racecar"` → Output: `"racecar"`
- Input: `"abcde"` → Output: `"edcba"`

---

4. First Non-Repeating Character

Problem Statement:

Find and return the first non-repeating character in a string.

Test Cases:

- Input: `"swiss"` → Output: `'w'`
- Input: `"racecars"` → Output: `'e'`
- Input: `"aabb"` → Output: `None`
- Input: `"apple"` → Output: `'a'`

---

5. Check if Two Strings are Anagrams

Problem Statement:

Two strings are anagrams if they contain the same characters in a different order.

Test Cases:

- Input: `"listen", "silent"` → Output: `true`
- Input: `"hello", "world"` → Output: `false`
- Input: `"rat", "tar"` → Output: `true`
- Input: `"abcd", "abc"` → Output: `false`

---

6. Count Occurrences of a Character

Problem Statement:

Count how many times a specific character appears in a string.

Test Cases:

- Input: `"hello", 'l'` → Output: `2`

- Input: `"banana", 'a'` → Output: `3`
- Input: `"aaaaa", 'a'` → Output: `5`
- Input: `"abcdef", 'z'` → Output: `0`

---

7. Convert String to Uppercase and Lowercase

Problem Statement:

Write a program that converts a given string to uppercase and lowercase.

Test Cases:

- Input: `"Java"` → Output: `Uppercase: "JAVA", Lowercase: "java"`
- Input: `"HELLO"` → Output: `Uppercase: "HELLO", Lowercase: "hello"`
- Input: `"world"` → Output: `Uppercase: "WORLD", Lowercase: "world"`
- Input: `"MiXeD"` → Output: `Uppercase: "MIXED", Lowercase: "mixed"`

---

8. Replace Character in a String

Problem Statement:

Replace all occurrences of a specific character in a string.

Test Cases:

- Input: `"hello", 'l', 'p'` → Output: `"heppo"`
- Input: `"banana", 'a', 'o'` → Output: `"bonono"`
- Input: `"test", 't', 'x'` → Output: `"xesx"`
- Input: `"abcdef", 'z', 'y'` → Output: `"abcdef"` (No change)

---

9. Find String Length without `length()` Method

Problem Statement:

Find the length of a given string without using the built-in `length()` method.

Test Cases:

- Input: `"hello"` → Output: `5`
- Input: `""` → Output: `0`
- Input: `"Java"` → Output: `4`
- Input: `"ABCDE"` → Output: `5`

---

10. Remove Whitespace from a String

Problem Statement:

Remove all white spaces from a given string.

Test Cases:

- Input: `" Hello World "` → Output: `"HelloWorld"`
- Input: `"a b c"` → Output: `"abc"`
- Input: `" 123 "` → Output: `"123"`
- Input: `"NoSpaces"` → Output: `"NoSpaces"`

Intermediate Level Questions

11. Check if String Contains Only Digits

Problem Statement:

Check if a given string consists only of digits.

Test Cases:

- Input: `"12345"` → Output: `true`
- Input: `"12a45"` → Output: `false`
- Input: `"007"` → Output: `true`
- Input: `""` → Output: `false`

---

12. Split a String into Words

Problem Statement:

Write a program to split a string into an array of words.

Test Cases:

- Input: `"Java is fun"` → Output: `["Java", "is", "fun"]`
- Input: `" hello world "` → Output: `["hello", "world"]`
- Input: `"one-word"` → Output: `["one-word"]`
- Input: `""` → Output: `[]`

---

13. Capitalize First Letter of Each Word

Problem Statement:

Write a function to capitalize the first letter of each word.

Test Cases:

- Input: `"hello world"` → Output: `"Hello World"`
- Input: `"java is cool"` → Output: `"Java Is Cool"`
- Input: `" hello "` → Output: `"Hello"`
- Input: `"ALL CAPS"` → Output: `"All Caps"`

---

14. Remove Duplicate Characters from a String

Problem Statement:

Write a program to remove duplicate characters from a string.

Test Cases:

- Input: `"banana"` → Output: `"ban"`
- Input: `"aabbcc"` → Output: `"abc"`
- Input: `"hello"` → Output: `"helo"`
- Input: `"abcdef"` → Output: `"abcdef"`

---

15. Find All Substrings of a String

Problem Statement:

Generate and print all substrings of a given string.

Test Cases:

- Input: `"abc"` → Output: `["a", "b", "c", "ab", "bc", "abc"]`
- Input: `"a"` → Output: `["a"]`
- Input: `""` → Output: `[]`
- Input: `"xy"` → Output: `["x", "y", "xy"]`

16. Reverse Words in a Sentence

Problem Statement:

Write a Java program that reverses the order of words in a given sentence.

Test Cases:

- Input: `"Hello World"` → Output: `"World Hello"`
- Input: `"Java is fun"` → Output: `"fun is Java"`
- Input: `"a b c"` → Output: `"c b a"`
- Input: `"single"` → Output: `"single"`

---

17. Longest Common Prefix

Problem Statement:

Find the longest common prefix among an array of strings.

Test Cases:

- Input: `["flower", "flow", "flight"]` → Output: `"fl"`
- Input: `["dog", "racecar", "car"]` → Output: `""`
- Input: `["apple", "appetizer", "application"]` → Output: `"app"`
- Input: `["java", "javac", "javascript"]` → Output: `"java"`

---

18. Check if a String is a Rotation of Another

Problem Statement:

Check if one string is a rotation of another using a single substring operation.

Test Cases:

- Input: `"waterbottle", "erbottlewat"` → Output: `true`
- Input: `"hello", "lohel"` → Output: `true`
- Input: `"hello", "ohlle"` → Output: `false`
- Input: `"abc", "cab"` → Output: `true`

---

19. Find the Most Frequent Character

Problem Statement:

Find the most frequently occurring character in a given string.

Test Cases:

- Input: `"hello"` → Output: `'l'`
- Input: `"banana"` → Output: `'a'`
- Input: `"abcdef"` → Output: `'a'` (All are equal)
- Input: `"mississippi"` → Output: `'i'`

---

20. Check if String is a Pangram

Problem Statement:

A pangram is a sentence that contains every letter of the alphabet at least once. Write a program to check if a given string is a pangram.

Test Cases:

- Input: `"The quick brown fox jumps over the lazy dog"` → Output: `true`
- Input: `"Hello World"` → Output: `false`
- Input: `"Pack my box with five dozen liquor jugs"` → Output: `true`
- Input: `"abcdefghijklmnopqrstuvwxyz"` → Output: `true`

---

21. Reverse a String Using Recursion

Problem Statement:

Write a recursive function to reverse a string.

Test Cases:

- Input: `"hello"` → Output: `"olleh"`
- Input: `"Java"` → Output: `"avaJ"`
- Input: `"racecar"` → Output: `"racecar"`
- Input: `"abcde"` → Output: `"edcba"`

---

22. Convert Integer to String Without `toString()`

Problem Statement:

Convert an integer to a string without using `toString()`.

Test Cases:

- Input: `1234` → Output: `"1234"`
- Input: `-567` → Output: `"-567"`
- Input: `0` → Output: `"0"`
- Input: `789` → Output: `"789"`

---

23. Find the Longest Palindromic Substring

Problem Statement:

Find the longest palindromic substring in a given string.

Test Cases:

- Input: `"babad"` → Output: `"bab"` or `"aba"`
- Input: `"cbbd"` → Output: `"bb"`
- Input: `"racecar"` → Output: `"racecar"`
- Input: `"a"` → Output: `"a"`

24. Remove All Adjacent Duplicates

Problem Statement:

Given a string, remove all adjacent duplicate characters.

Test Cases:

- Input: `"aabbcc"` → Output: `""`
- Input: `"abccba"` → Output: `""`
- Input: `"mississippi"` → Output: `"m"`
- Input: `"abc"` → Output: `"abc"`

---

25. Count Words in a String

Problem Statement:

Write a function to count the number of words in a given string.

Test Cases:

- Input: `"Hello world"` → Output: `2`
- Input: `"Java is fun"` → Output: `3`
- Input: `" multiple spaces "` → Output: `2`
- Input: `""` → Output: `0`

Advanced Level Questions

26. Find All Permutations of a String

Problem Statement:

Generate all possible permutations of a given string.

Test Cases:

- Input: `"abc"` → Output: `["abc", "acb", "bac", "bca", "cab", "cba"]`
- Input: `"ab"` → Output: `["ab", "ba"]`
- Input: `"a"` → Output: `["a"]`
- Input: `""` → Output: `[]`

---

27. String Compression (Run-Length Encoding)

Problem Statement:

Compress a string using run-length encoding (e.g., `"aaabb"` → `"a3b2"`).

Test Cases:

- Input: `"aaabb"` → Output: `"a3b2"`
- Input: `"abc"` → Output: `"a1b1c1"`
- Input: `"aabbcc"` → Output: `"a2b2c2"`
- Input: `"aaaaa"` → Output: `"a5"`

---

28. Convert Roman Numeral to Integer

Problem Statement:

Convert a Roman numeral string to an integer.

Test Cases:

- Input: `"III"` → Output: `3`
- Input: `"IX"` → Output: `9`
- Input: `"LVIII"` → Output: `58`

- Input: `"MCMXCIV"` → Output: `1994`

---

29. Find the Minimum Window Substring

Problem Statement:

Find the smallest substring that contains all characters of another string.

Test Cases:

- Input: `s = "ADOBECODEBANC", t = "ABC"` → Output: `"BANC"`
- Input: `s = "a", t = "a"` → Output: `"a"`
- Input: `s = "a", t = "aa"` → Output: `""`
- Input: `s = "abc", t = "ac"` → Output: `"abc"`

---

30. Edit Distance (Levenshtein Distance)

Problem Statement:

Find the minimum number of operations (insertions, deletions, or replacements) to convert one string into another.

Test Cases:

- Input: `"horse", "ros"` → Output: `3`
- Input: `"intention", "execution"` → Output: `5`
- Input: `"abcd", "ab"` → Output: `2`
- Input: `"abcdef", "azced"` → Output: `3`