️ **PRACTICE PROBLEM 1:**

**String Creation and Manipulation**

**Task**: Create a program that demonstrates different ways to create strings and basic manipulation.

public class StringManipulation{

public static void main(String[] args) {

// TODO: Create the same string "Java Programming" using 3 different methods:

// 1. String literal

// 2. new String() constructor

// 3. Character array

// TODO: Compare the strings using == and .equals()

// Print the results and explain the difference

// TODO: Create a string with escape sequences that displays: // Programming Quote:

// "Code is poetry" - Unknown

// Path: C:\Java\Projects

}

} l

public class StringManipulation {

public static void main(String[] args) {

// 1. String literal

String str1 = "Java Programming";

// 2. new String() constructor

String str2 = new String("Java Programming");

// 3. Character array

char[] charArray = {'J', 'a', 'v', 'a', ' ', 'P', 'r', 'o', 'g', 'r', 'a', 'm', 'm', 'i', 'n', 'g'};

String str3 = new String(charArray);

// Displaying all three strings

System.out.println("String 1 (Literal): " + str1);

System.out.println("String 2 (Using new String() constructor): " + str2);

System.out.println("String 3 (From Character array): " + str3);

// Comparing strings using == and .equals()

System.out.println("\nComparing using == operator:");

System.out.println("str1 == str2: " + (str1 == str2));

System.out.println("str1 == str3: " + (str1 == str3));

System.out.println("\nComparing using .equals() method:");

System.out.println("str1.equals(str2): " + str1.equals(str2));

System.out.println("str1.equals(str3): " + str1.equals(str3));

// Creating a string with escape sequences

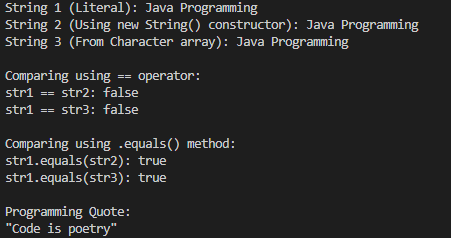
String quote = "Programming Quote:\n\"Code is poetry\"";

// Printing the string with escape sequences

System.out.println("\n" + quote);

}

}

Output:  


**PRACTICE PROBLEM 2:**

**String Input and Processing**

**Task**: Create a program that takes user input and processes it using various string methods.

import java.util.Scanner;

public class StringMethods{

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// TODO: Ask user for their full name (first and last name) // TODO: Ask user for their favorite programming language

// TODO: Ask user for a sentence about their programming experience

// TODO: Process the input:

// 1. Extract first and last name separately

// 2. Count total characters in the sentence (excluding spaces) // 3. Convert programming language to uppercase

// 4. Display a formatted summary

scanner.close();

}

}

import java.util.Scanner;

public class StringMethods {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Step 1: Ask user for their full name (first and last name)

System.out.print("Enter your full name (first and last): ");

String fullName = scanner.nextLine();

// Step 2: Ask user for their favorite programming language

System.out.print("Enter your favorite programming language: ");

String language = scanner.nextLine();

// Step 3: Ask user for a sentence about their programming experience

System.out.print("Describe your programming experience: ");

String experience = scanner.nextLine();

// Step 4: Process the input

// Extract first and last name separately

String[] nameParts = fullName.split(" ");

String firstName = nameParts[0];

String lastName = nameParts.length > 1 ? nameParts[1] : "";

// Count total characters in the sentence (excluding spaces)

int charCount = experience.replace(" ", "").length();

// Convert programming language to uppercase

String upperLanguage = language.toUpperCase();

// Step 5: Display a formatted summary

System.out.println("\n----- Summary -----");

System.out.println("First Name: " + firstName);

System.out.println("Last Name: " + lastName);

System.out.println("Favorite Programming Language (uppercase): " + upperLanguage);

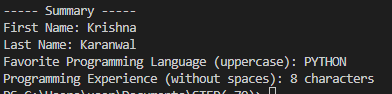
System.out.println("Programming Experience (without spaces): " + charCount + " characters");

scanner.close();

}

}

Output:



**PRACTICE PROBLEM 3:**

**String Arrays and Methods**

**Task**: Create a program that manages a list of student names using string arrays and methods.

public class StringArrays {

// TODO: Create a method that takes a string array of names // and returns the longest name

public static String findLongestName(String[] names) {

// Your code here

}

// TODO: Create a method that counts how many names

// start with a given letter (case-insensitive)

public static int countNamesStartingWith(String[] names, char letter) { // Your code here

}

// TODO: Create a method that formats all names to "Last, First" format // Assume names are given as "First Last"

public static String[] formatNames(String[] names) {

// Your code here

}

public static void main(String[] args) {

String[] students = {"John Smith", "Alice Johnson", "Bob Brown", "Carol Davis", "David Wilson"};

// TODO: Test all your methods and display results

}

}

public class StringArrays {

// Method to find the longest name

public static String findLongestName(String[] names) {

String longestName = names[0];

for (String name : names) {

if (name.length() > longestName.length()) {

longestName = name;

}

}

return longestName;

}

// Method to count how many names start with a given letter (case-insensitive)

public static int countNamesStartingWith(String[] names, char letter) {

int count = 0;

for (String name : names) {

if (Character.toLowerCase(name.charAt(0)) == Character.toLowerCase(letter)) {

count++;

}

}

return count;

}

// Method to format all names to "Last, First" format (assumes names are "First Last")

public static String[] formatNames(String[] names) {

String[] formattedNames = new String[names.length];

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

String[] nameParts = names[i].split(" ");

formattedNames[i] = nameParts[1] + ", " + nameParts[0];

}

return formattedNames;

}

public static void main(String[] args) {

String[] students = {"Krishna Karanwal", "Hamsini Goel", "Anuj Vijay", "Aayush Rai",

"Shubhi Vijay"};

// Test all methods and display results

System.out.println("Longest name: " + findLongestName(students));

System.out.println("Names starting with 'A': " + countNamesStartingWith(students, 'A'));

// Format names and display them

String[] formattedNames = formatNames(students);

System.out.println("\nFormatted names:");

for (String name : formattedNames) {

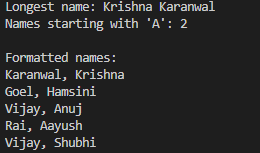
System.out.println(name);

}

}

}

Output:



️ **PRACTICE PROBLEM 4:**

**Complete String Application (10 minutes)**

**Task**: Create a simple text processor that combines all concepts learned. import java.util.Scanner;

public class TextProcessor{

// TODO: Method to clean and validate input

public static String cleanInput(String input) {

// Remove extra spaces, convert to proper case

// Return cleaned string

}

// TODO: Method to analyze text

public static void analyzeText(String text) {

// Count: words, sentences, characters

// Find: longest word, most common character

// Display statistics

}

// TODO: Method to create word array and sort alphabetically public static String[] getWordsSorted(String text) { // Split text into words, remove punctuation, sort

// Return sorted array

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// TODO: Create a text processor that:

// 1. Asks user for a paragraph of text

// 2. Cleans and validates the input

// 3. Analyzes the text (word count, character count, etc.) // 4. Shows the words in alphabetical order

// 5. Allows user to search for specific words

System.out.println("=== TEXT PROCESSOR ===");

// Your implementation here

scanner.close();

}

}

import java.util.\*;

public class TextProcessor {

// Method to clean and validate input (removes extra spaces, converts to proper case)

public static String cleanInput(String input) {

// Remove leading/trailing spaces, and reduce multiple spaces to one

input = input.trim().replaceAll("\\s+", " ");

// Capitalize the first letter of each word

String[] words = input.split(" ");

StringBuilder cleaned = new StringBuilder();

for (String word : words) {

cleaned.append(Character.toUpperCase(word.charAt(0)))

.append(word.substring(1).toLowerCase())

.append(" ");

}

return cleaned.toString().trim();

}

// Method to analyze text (word count, sentence count, character count, etc.)

public static void analyzeText(String text) {

// Count words, sentences, and characters

String[] words = text.split(" ");

int wordCount = words.length;

// Split sentences by periods, exclamation marks, and question marks

String[] sentences = text.split("[.!?]");

int sentenceCount = sentences.length;

// Count characters (excluding spaces)

int charCount = text.replace(" ", "").length();

// Find the longest word

String longestWord = Arrays.stream(words)

.max(Comparator.comparingInt(String::length))

.orElse("");

// Find the most common character

Map<Character, Integer> charFreq = new HashMap<>();

for (char c : text.toCharArray()) {

if (Character.isLetter(c)) {

c = Character.toLowerCase(c);

charFreq.put(c, charFreq.getOrDefault(c, 0) + 1);

}

}

char mostCommonChar = charFreq.entrySet()

.stream()

.max(Map.Entry.comparingByValue())

.map(Map.Entry::getKey)

.orElse(' ');

// Display the statistics

System.out.println("\n--- Text Analysis ---");

System.out.println("Word count: " + wordCount);

System.out.println("Sentence count: " + sentenceCount);

System.out.println("Character count (excluding spaces): " + charCount);

System.out.println("Longest word: " + longestWord);

System.out.println("Most common character: " + mostCommonChar);

}

// Method to create a word array and sort alphabetically

public static String[] getWordsSorted(String text) {

// Split text into words and remove punctuation

String[] words = text.replaceAll("[^a-zA-Z ]", "").split("\\s+");

Arrays.sort(words);

return words;

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Ask user for a paragraph of text

System.out.println("=== TEXT PROCESSOR ===");

System.out.print("Enter a paragraph of text: ");

String inputText = scanner.nextLine();

// Step 1: Clean and validate the input

String cleanedText = cleanInput(inputText);

System.out.println("\nCleaned Text: " + cleanedText);

// Step 2: Analyze the text (word count, character count, etc.)

analyzeText(cleanedText);

// Step 3: Show the words in alphabetical order

String[] sortedWords = getWordsSorted(cleanedText);

System.out.println("\nWords in alphabetical order:");

for (String word : sortedWords) {

System.out.println(word);

}

// Step 4: Allow user to search for specific words

System.out.print("\nEnter a word to search for: ");

String searchWord = scanner.nextLine();

if (cleanedText.contains(searchWord)) {

System.out.println("The word \"" + searchWord + "\" is found in the text.");

} else {

System.out.println("The word \"" + searchWord + "\" is not found in the text.");

}

scanner.close();

}

}

Output: