# Java Programs for QA Interview (5+ Years Experience)

## 1. Core Java Logic-Based Programs

* • Reverse string without using built-in methods
* • Check if string or number is palindrome
* • Find and print duplicate characters in a string
* • Check if two strings are anagrams
* • Count occurrences of words/characters in a string
* • Remove white spaces without using trim() or replaceAll()
* • Prime number, Fibonacci series, Armstrong number
* • Sort array without using Arrays.sort()
* • Find largest/smallest number in an array
* • Swap two numbers without third variable
* • Factorial (Recursive and Iterative)
* • Print star, triangle, and pyramid patterns

## 2. Collections & Data Structures

* • Iterate, add/remove, and sort ArrayList, LinkedList, HashMap
* • Use HashMap to count frequencies and manipulate key-values
* • Understand Set vs List behavior (Uniqueness, nulls, duplicates)
* • Use TreeMap/TreeSet for auto-sorting
* • Sort Map by Key or Value using Comparator or Stream API
* • Sort list of custom objects (e.g. Employee by age/salary)

## 3. Java + Selenium/Automation

* • Read/write Excel using Apache POI
* • JSON Parsing using Jackson/Gson
* • Read/write to text or CSV files
* • WebDriver setup and basic actions (open browser, click, sendKeys, waits)
* • Use of Selenium Locators (XPath/CSS)
* • Create Page Object Model structure (base page, login page etc.)
* • API Automation using RestAssured (GET/POST with validations)
* • Add logging using Log4j/SLF4J
* • Load config data using Java Properties File

## 4. OOPs & Concept-Based Programs

* • Parent-child class with method overriding (Inheritance)
* • Compile-time vs Runtime examples (Polymorphism)
* • POJO class with private fields + getters/setters (Encapsulation)
* • Interface vs Abstract class example (Abstraction)
* • Constructor Chaining using this() and super()
* • Count instances or utility methods using static keyword

## 5. Practical Programs for QA Rounds

* • Read test data from Excel and run test logic
* • Store and fetch test data using Map
* • Compare two lists (actual vs expected) for assertions
* • Retry failed test logic using TestNG or Java logic
* • Parse string from API response and extract data
* • Convert JSON to POJO and validate fields