---------- **SDET Coding Questions** ----------

◆ Java & Data Structures

1. Reverse a string without using reverse()

```
public class ReverseString {
    public static void main(String[] args) {
        String input = "hello";
        String reversed = "";
        for (int i = input.length() - 1; i >= 0; i--) {
            reversed += input.charAt(i);
        }
        System.out.println("Reversed: " + reversed);
    }
}
```

---

2. Check if a string is a palindrome

```
public class PalindromeCheck {
    public static void main(String[] args) {
        String input = "madam";
        String reversed = new StringBuilder(input).reverse().toString();
        System.out.println("Is palindrome: " + input.equals(reversed));
    }
}
```

---

3. Find the first non-repeating character in a string

```
import java.util.*;
public class FirstNonRepeatingChar {
    public static void main(String[] args) {
        String input = "aabbcdeff";
        Map<Character, Integer> map = new LinkedHashMap<>();
        for (char c : input.toCharArray()) {
            map.put(c, map.getOrDefault(c, 0) + 1);
        }
        for (char c : map.keySet()) {
            if (map.get(c) == 1) {
                System.out.println("First non-repeating: " + c);
                break;
            }
        }
    }
}
```

---

4. Count frequency of each character in a string

```
import java.util.*;
public class CharFrequency {
    public static void main(String[] args) {
        String input = "aabbccdde";
        Map<Character, Integer> freq = new HashMap<>();
        for (char c : input.toCharArray()) {
            freq.put(c, freq.getOrDefault(c, 0) + 1);
        }
        System.out.println(freq);
    }
}
```

---

5. Remove duplicate characters from a string

```
public class RemoveDuplicatesFromString {
    public static void main(String[] args) {
        String input = "aabbccde";
        StringBuilder result = new StringBuilder();
        Set<Character> seen = new HashSet<>();
        for (char c : input.toCharArray()) {
            if (!seen.contains(c)) {
                seen.add(c);
                result.append(c);
            }
        }
```

```
        System.out.println("Without duplicates: " + result.toString());
    }
}
```

---

6. Check if two strings are anagrams

```
import java.util.*;
public class AnagramCheck {
    public static void main(String[] args) {
        String s1 = "listen";
        String s2 = "silent";
        char[] arr1 = s1.toCharArray();
        char[] arr2 = s2.toCharArray();
        Arrays.sort(arr1);
        Arrays.sort(arr2);
        System.out.println("Are anagrams: " + Arrays.equals(arr1, arr2));
    }
}
```

---

7. Sort an array using bubble sort

```
import java.util.*;
public class BubbleSort {
    public static void main(String[] args) {
        int[] arr = {5, 2, 9, 1};
        for (int i = 0; i < arr.length - 1; i++) {
            for (int j = 0; j < arr.length - 1 - i; j++) {
                if (arr[j] > arr[j + 1]) {
                    int temp = arr[j];
                    arr[j] = arr[j + 1];
                    arr[j + 1] = temp;
                }
            }
        }
        System.out.println(Arrays.toString(arr));
    }
}
```

---

8. Sort an array using selection sort

```
import java.util.*;
public class SelectionSort {
    public static void main(String[] args) {
        int[] arr = {5, 2, 9, 1};
        for (int i = 0; i < arr.length - 1; i++) {
            int minIndex = i;
            for (int j = i + 1; j < arr.length; j++) {
                if (arr[j] < arr[minIndex]) {
                    minIndex = j;
                }
            }
            int temp = arr[minIndex];
            arr[minIndex] = arr[i];
            arr[i] = temp;
        }
        System.out.println(Arrays.toString(arr));
    }
}
```

---

9. Find the second largest number in an array

```
public class SecondLargest {
    public static void main(String[] args) {
        int[] arr = {10, 20, 4, 45, 99};
        int first = Integer.MIN_VALUE;
        int second = Integer.MIN_VALUE;
        for (int num : arr) {
            if (num > first) {
                second = first;
                first = num;
            } else if (num > second && num != first) {
```

```
                second = num;
            }
        }
        System.out.println("Second largest: " + second);
    }
}
```

---

10. Find the missing number from an array of 1 to n

```
public class MissingNumber {
    public static void main(String[] args) {
        int[] arr = {1, 2, 4, 5, 6};
        int n = 6;
        int sum = n * (n + 1) / 2;
        for (int num : arr) {
            sum -= num;
        }
        System.out.println("Missing number: " + sum);
    }
}
```

---

11. Find all pairs in an array with a given sum

```
public class PairWithSum {
    public static void main(String[] args) {
        int[] arr = {2, 4, 3, 5, 7};
        int target = 7;
        for (int i = 0; i < arr.length; i++) {
            for (int j = i + 1; j < arr.length; j++) {
                if (arr[i] + arr[j] == target) {
                    System.out.println("Pair: (" + arr[i] + ", " + arr[j] + ")");
                }
            }
        }
    }
}
```

---

12. Remove duplicates from an array

```
import java.util.*;
public class RemoveDuplicatesArray {
    public static void main(String[] args) {
        int[] arr = {1, 2, 2, 3, 4, 4, 5};
        Set<Integer> set = new LinkedHashSet<>();
        for (int num : arr) {
            set.add(num);
        }
        System.out.println("Array without duplicates: " + set);
    }
}
```

---

13. Reverse each word in a sentence

```
public class ReverseWordsInSentence {
    public static void main(String[] args) {
        String input = "Hello World";
        String[] words = input.split(" ");
        StringBuilder result = new StringBuilder();
        for (String word : words) {
            result.append(new StringBuilder(word).reverse()).append(" ");
        }
        System.out.println("Reversed words: " + result.toString().trim());
    }
}
```

---

14. Convert Roman numerals to integer

```
import java.util.*;
public class RomanToInteger {
    public static void main(String[] args) {
        String roman = "IX";
        Map<Character, Integer> map = new HashMap<>();
```

```
        map.put('I', 1);
        map.put('V', 5);
        map.put('X', 10);
        map.put('L', 50);
        map.put('C', 100);
        map.put('D', 500);
        map.put('M', 1000);
        int sum = 0;
        for (int i = 0; i < roman.length(); i++) {
            if (i > 0 && map.get(roman.charAt(i)) > map.get(roman.charAt(i - 1))) {
                sum += map.get(roman.charAt(i)) - 2 * map.get(roman.charAt(i - 1));
            } else {
                sum += map.get(roman.charAt(i));
            }
        }
        System.out.println("Integer: " + sum);
    }
}
```

---

15. Convert integer to Roman numerals

```
public class IntegerToRoman {
    public static void main(String[] args) {
        int num = 58;
        int[] values = {1000, 900, 500, 400, 100, 90, 50, 40, 10, 9, 5, 4, 1};
        String[] symbols = {"M", "CM", "D", "CD", "C", "XC", "L", "XL", "X", "IX", "V", "IV", "I"};
        StringBuilder roman = new StringBuilder();
        for (int i = 0; i < values.length; i++) {
            while (num >= values[i]) {
                num -= values[i];
                roman.append(symbols[i]);
            }
        }
        System.out.println("Roman: " + roman.toString());
    }
}
```

---

16. Merge two sorted arrays into one sorted array

```
import java.util.*;
public class MergeSortedArrays {
    public static void main(String[] args) {
        int[] a = {1, 3, 5};
        int[] b = {2, 4, 6};
        int[] merged = new int[a.length + b.length];
        int i = 0, j = 0, k = 0;
        while (i < a.length && j < b.length) {
            if (a[i] < b[j]) {
                merged[k++] = a[i++];
            } else {
                merged[k++] = b[j++];
            }
        }
        while (i < a.length) {
            merged[k++] = a[i++];
        }
        while (j < b.length) {
            merged[k++] = b[j++];
        }
        System.out.println(Arrays.toString(merged));
    }
}
```

---

17. Implement binary search

```
public class BinarySearch {
    public static void main(String[] args) {
        int[] arr = {1, 3, 5, 7, 9};
        int target = 5;
        int left = 0, right = arr.length - 1;
```

```
        while (left <= right) {
            int mid = left + (right - left) / 2;
            if (arr[mid] == target) {
                System.out.println("Found at index: " + mid);
                return;
            } else if (arr[mid] < target) {
                left = mid + 1;
            } else {
                right = mid - 1;
            }
        }
        System.out.println("Not found");
    }
}
```

---

18. Find longest common prefix among an array of strings

```
public class LongestCommonPrefix {
    public static void main(String[] args) {
        String[] strs = {"flower", "flow", "flight"};
        if (strs == null || strs.length == 0) {
            System.out.println("");
            return;
        }
        String prefix = strs[0];
        for (int i = 1; i < strs.length; i++) {
            while (strs[i].indexOf(prefix) != 0) {
                prefix = prefix.substring(0, prefix.length() - 1);
                if (prefix.isEmpty()) {
                    System.out.println("");
                    return;
                }
            }
        }
        System.out.println("Longest Common Prefix: " + prefix);
    }
}
```

---

19. Count vowels and consonants in a string

```
public class VowelConsonantCount {
    public static void main(String[] args) {
        String input = "Hello World".toLowerCase();
        int vowels = 0, consonants = 0;
        for (char c : input.toCharArray()) {
            if (Character.isLetter(c)) {
                if ("aeiou".indexOf(c) != -1) vowels++;
                else consonants++;
            }
        }
        System.out.println("Vowels: " + vowels + ", Consonants: " + consonants);
    }
}
```

---

20. Move all zeros to the end of the array

```
import java.util.*;
public class MoveZerosToEnd {
    public static void main(String[] args) {
        int[] arr = {0, 1, 0, 3, 12};
        int index = 0;
        for (int num : arr) {
            if (num != 0) {
                arr[index++] = num;
            }
        }
        while (index < arr.length) {
            arr[index++] = 0;
        }
        System.out.println(Arrays.toString(arr));
```

```
    }
}
```

---

21. Rotate an array to the right by k steps

```
public class RotateArray {
    public static void main(String[] args) {
        int[] arr = {1,2,3,4,5,6};
        int k = 2;
        int n = arr.length;
        k = k % n;
        reverse(arr, 0, n - 1);
        reverse(arr, 0, k - 1);
        reverse(arr, k, n - 1);
        for (int num : arr) System.out.print(num + " ");
    }
    static void reverse(int[] arr, int start, int end) {
        while (start < end) {
            int temp = arr[start];
            arr[start++] = arr[end];
            arr[end--] = temp;
        }
    }
}
```

---

22. First repeated character in a string

```
public class FirstRepeatedChar {
    public static void main(String[] args) {
        String str = "programming";
        Set<Character> set = new HashSet<>();
        for (char c : str.toCharArray()) {
            if (!set.add(c)) {
                System.out.println("First repeated: " + c);
                break;
            }
        }
    }
}
```

---

23. Sort characters by frequency

```
public class FrequencySort {
    public static void main(String[] args) {
        String s = "tree";
        Map<Character, Integer> freq = new HashMap<>();
        for (char c : s.toCharArray()) freq.put(c, freq.getOrDefault(c, 0) + 1);
        List<Character> sorted = new ArrayList<>(freq.keySet());
        sorted.sort((a, b) -> freq.get(b) - freq.get(a));
        StringBuilder sb = new StringBuilder();
        for (char c : sorted) sb.append(String.valueOf(c).repeat(freq.get(c)));
        System.out.println(sb.toString());
    }
}
```

---

24. Stack using array

```
public class StackUsingArray {
    static class Stack {
        int[] arr = new int[5];
        int top = -1;
        void push(int val) {
            if (top < arr.length - 1) arr[++top] = val;
        }
        int pop() {
            return top >= 0 ? arr[top--] : -1;
        }
    }
    public static void main(String[] args) {
        Stack s = new Stack();
        s.push(10); s.push(20); System.out.println(s.pop());
```

```
    }
}
```

---

25. Queue using LinkedList

```
public class QueueUsingLinkedList {
    public static void main(String[] args) {
        Queue<Integer> q = new LinkedList<>();
        q.add(1); q.add(2);
        System.out.println(q.poll());
    }
}
```

---

26. Longest substring without repeating characters

```
public class LongestUniqueSubstring {
    public static void main(String[] args) {
        String s = "abcabcbb";
        Set<Character> set = new HashSet<>();
        int left = 0, max = 0;
        for (int right = 0; right < s.length(); right++) {
            while (!set.add(s.charAt(right)))
                set.remove(s.charAt(left++));
            max = Math.max(max, right - left + 1);
        }
        System.out.println("Max length: " + max);
    }
}
```

---

27. Find common elements in two arrays

```
public class CommonElements {
    public static void main(String[] args) {
        int[] a = {1, 2, 3, 4}, b = {3, 4, 5, 6};
        Set<Integer> set = new HashSet<>();
        for (int x : a) set.add(x);
        for (int y : b) if (set.contains(y)) System.out.print(y + " ");
    }
}
```

---

28. Count number of words in sentence

```
public class WordCount {
    public static void main(String[] args) {
        String sentence = "Hello from the other side";
        String[] words = sentence.trim().split("\\s+");
        System.out.println("Word count: " + words.length);
    }
}
```

---

29. Count digit frequency in a number

```
public class DigitFrequency {
    public static void main(String[] args) {
        int num = 112345211;
        int[] freq = new int[10];
        while (num > 0) {
            freq[num % 10]++;
            num /= 10;
        }
        for (int i = 0; i < 10; i++) if (freq[i] > 0)
            System.out.println(i + ": " + freq[i]);
    }
}
```

---

30. Perfect square check (without Math.sqrt)

```
public class PerfectSquareCheck {
    public static void main(String[] args) {
        int n = 49;
        int i = 1;
        while (i * i <= n) {
            if (i * i == n) {
```

```
                System.out.println("Perfect square");
                return;
            }
            i++;
        }
        System.out.println("Not a perfect square");
    }
}
```

---

31. Generate Fibonacci series up to n terms

```
public class FibonacciSeries {
    public static void main(String[] args) {
        int n = 10, a = 0, b = 1;
        System.out.print(a + " " + b);
        for (int i = 2; i < n; i++) {
            int next = a + b;
            System.out.print(" " + next);
            a = b;
            b = next;
        }
    }
}
```

---

32. Check if an array is sorted in ascending order

```
public class IsArraySorted {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 5, 6};
        boolean sorted = true;
        for (int i = 1; i < arr.length; i++) {
            if (arr[i] < arr[i - 1]) {
                sorted = false;
                break;
            }
        }
        System.out.println("Is sorted: " + sorted);
    }
}
```

---

33. Check if a string has balanced brackets

```
import java.util.Stack;

public class BalancedBrackets {
    public static void main(String[] args) {
        String str = "{[()]}";
        Stack<Character> stack = new Stack<>();
        for (char ch : str.toCharArray()) {
            if ("{[(".contains(String.valueOf(ch))) {
                stack.push(ch);
            } else if ("}])".contains(String.valueOf(ch))) {
                if (stack.isEmpty() || !isMatch(stack.pop(), ch)) {
                    System.out.println("Not balanced");
                    return;
                }
            }
        }
        System.out.println(stack.isEmpty() ? "Balanced" : "Not balanced");
    }

    private static boolean isMatch(char open, char close) {
        return (open == '{' && close == '}') ||
               (open == '[' && close == ']') ||
               (open == '(' && close == ')');
    }
}
```

---

34. Find GCD of two numbers

```
public class FindGCD {
   public static void main(String[] args) {
      int a = 36, b = 60;
      while (b != 0) {
         int temp = b;
         b = a % b;
         a = temp;
      }
      System.out.println("GCD: " + a);
   }
}
```

---

35. Count and print duplicate elements in an array

```
import java.util.*;

public class DuplicateElements {
   public static void main(String[] args) {
      int[] arr = {1, 2, 3, 2, 3, 4, 5};
      Map<Integer, Integer> map = new HashMap<>();
      for (int num : arr) map.put(num, map.getOrDefault(num, 0) + 1);
      for (Map.Entry<Integer, Integer> entry : map.entrySet())
         if (entry.getValue() > 1)
            System.out.println("Duplicate: " + entry.getKey());
   }
}
```

---

36. Sort a list of integers using Java Collections

```
import java.util.*;

public class SortList {
   public static void main(String[] args) {
      List<Integer> list = Arrays.asList(5, 3, 8, 1);
      Collections.sort(list);
      System.out.println(list);
   }
}
```

---

37. Find factorial using recursion

```
public class FactorialRecursion {
   public static void main(String[] args) {
      int n = 5;
      System.out.println("Factorial: " + factorial(n));
   }

   static int factorial(int n) {
      return (n == 0) ? 1 : n * factorial(n - 1);
   }
}
```

---

38. Reverse a LinkedList without using Collections.reverse()

```
import java.util.LinkedList;

public class ReverseLinkedList {
   public static void main(String[] args) {
      LinkedList<Integer> list = new LinkedList<>();
      list.add(1); list.add(2); list.add(3);
      for (int i = list.size() - 1; i >= 0; i--)
         System.out.print(list.get(i) + " ");
   }
}
```

---

39. Remove all non-alphanumeric characters from a string

```
public class RemoveNonAlphanumeric {
   public static void main(String[] args) {
      String s = "He@#llo! 123";
      s = s.replaceAll("[^a-zA-Z0-9]", "");
```

```
        System.out.println(s);
    }
}
```

---

40. Convert a sentence to title case

```
public class TitleCase {
    public static void main(String[] args) {
        String sentence = "java is fun to learn";
        String[] words = sentence.split(" ");
        StringBuilder sb = new StringBuilder();
        for (String word : words)
            sb.append(Character.toUpperCase(word.charAt(0)))
              .append(word.substring(1)).append(" ");
        System.out.println(sb.toString().trim());
    }
}
```

🔹 Automation-Specific Logic (Java + Selenium)

---

1. Log in with username/password and validate successful login

```
public class LoginTest {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.get("https://example.com/login");
        driver.findElement(By.id("username")).sendKeys("testuser");
        driver.findElement(By.id("password")).sendKeys("password");
        driver.findElement(By.id("loginBtn")).click();
        boolean success = driver.findElement(By.id("welcomeMsg")).isDisplayed();
        System.out.println("Login successful: " + success);
        driver.quit();
    }
}
```

---

2. Select value from a dynamic dropdown

```
public class DynamicDropdown {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.get("https://example.com");
        List<WebElement> options = driver.findElements(By.cssSelector(".dropdown li"));
        for (WebElement option : options) {
            if (option.getText().equals("OptionValue")) {
                option.click();
                break;
            }
        }
        driver.quit();
    }
}
```

---

3. Handle stale element exception

```
public class HandleStaleElement {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.get("https://example.com");
        WebElement element = driver.findElement(By.id("refreshElement"));
        driver.navigate().refresh();
        try {
            element.click();
        } catch (StaleElementReferenceException e) {
            element = driver.findElement(By.id("refreshElement"));
            element.click();
        }
        driver.quit();
    }
}
```

4. Validate file download using Selenium

```
public class FileDownloadValidation {
   public static void main(String[] args) throws InterruptedException {
      WebDriver driver = new ChromeDriver();
      driver.get("https://example.com/download");
      driver.findElement(By.id("downloadBtn")).click();
      Thread.sleep(5000);
      File folder = new File("C:/Downloads");
      File[] files = folder.listFiles();
      boolean downloaded = Arrays.stream(files).anyMatch(f -> f.getName().contains("fileName"));
      System.out.println("Download success: " + downloaded);
      driver.quit();
   }
}
```

---

5. Automate file upload using Selenium

```
public class FileUpload {
   public static void main(String[] args) {
      WebDriver driver = new ChromeDriver();
      driver.get("https://example.com/upload");
      WebElement upload = driver.findElement(By.id("fileUpload"));
      upload.sendKeys("C:/path/to/file.txt");
      driver.findElement(By.id("submitBtn")).click();
      driver.quit();
   }
}
```

---

6. Capture all broken links on a webpage

```
public class BrokenLinksChecker {
   public static void main(String[] args) throws IOException {
      WebDriver driver = new ChromeDriver();
      driver.get("https://example.com");
      List<WebElement> links = driver.findElements(By.tagName("a"));
      for (WebElement link : links) {
         String url = link.getAttribute("href");
         if (url != null) {
            HttpURLConnection conn = (HttpURLConnection) new URL(url).openConnection();
            conn.setRequestMethod("HEAD");
            conn.connect();
            if (conn.getResponseCode() >= 400) {
               System.out.println("Broken link: " + url);
            }
         }
      }
      driver.quit();
   }
}
```

---

7. Take a screenshot only if test fails

```
public class ScreenshotOnFailure {
   public static void main(String[] args) {
      WebDriver driver = new ChromeDriver();
      try {
         driver.get("https://example.com");
         WebElement el = driver.findElement(By.id("notExist"));
         el.click();
      } catch (Exception e) {
         File scr = ((TakesScreenshot) driver).getScreenshotAs(OutputType.FILE);
         File dest = new File("screenshot.png");
         scr.renameTo(dest);
      } finally {
         driver.quit();
      }
   }
}
```

8. Custom retry logic in TestNG

```
public class RetryAnalyzer implements IRetryAnalyzer {
   int count = 0, maxRetry = 2;
   public boolean retry(ITestResult result) {
      if (count < maxRetry) {
         count++;
         return true;
      }
      return false;
   }
}
```

---

9. Dynamic XPath for changing ID

```
// Example: //*[@id='user_1234'] → //*[starts-with(@id, 'user_')]
By dynamicId = By.xpath("//*[starts-with(@id, 'user_')]");
```

---

10. Read data from Excel using Apache POI

```
public class ExcelReader {
   public static void main(String[] args) throws IOException {
      FileInputStream fis = new FileInputStream("data.xlsx");
      Workbook workbook = new XSSFWorkbook(fis);
      Sheet sheet = workbook.getSheetAt(0);
      for (Row row : sheet) {
         for (Cell cell : row) {
            System.out.print(cell.toString() + " ");
         }
         System.out.println();
      }
      workbook.close();
   }
}
```

---

11. Scroll to an element using JavaScriptExecutor

```
public class ScrollToElement {
   public static void main(String[] args) {
      WebDriver driver = new ChromeDriver();
      driver.get("https://example.com");
      WebElement element = driver.findElement(By.id("target"));
      ((JavascriptExecutor) driver).executeScript("arguments[0].scrollIntoView(true);", element);
      driver.quit();
   }
}
```

---

12. Handle JavaScript alert

```
public class HandleAlert {
   public static void main(String[] args) {
      WebDriver driver = new ChromeDriver();
      driver.get("https://example.com");
      driver.findElement(By.id("alertButton")).click();
      Alert alert = driver.switchTo().alert();
      alert.accept();
      driver.quit();
   }
}
```

---

13. Wait until element is visible using WebDriverWait

```
public class WaitUntilVisible {
   public static void main(String[] args) {
      WebDriver driver = new ChromeDriver();
      driver.get("https://example.com");
      WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
      WebElement el = wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("delayedElement")));
      el.click();
      driver.quit();
   }
}
```

14. Highlight element using JavaScriptExecutor

```
public class HighlightElement {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.get("https://example.com");
        WebElement element = driver.findElement(By.id("highlight"));
        JavascriptExecutor js = (JavascriptExecutor) driver;
        js.executeScript("arguments[0].style.border='3px solid red'", element);
        driver.quit();
    }
}
```

---

15. Capture console logs using Chrome DevTools

```
public class CaptureConsoleLogs {
    public static void main(String[] args) {
        ChromeOptions options = new ChromeOptions();
        options.setCapability("goog:loggingPrefs", Map.of("browser", Level.ALL));
        WebDriver driver = new ChromeDriver(options);
        driver.get("https://example.com");
        LogEntries logs = driver.manage().logs().get(LogType.BROWSER);
        logs.forEach(log -> System.out.println(log.getMessage()));
        driver.quit();
    }
}
```

---

16. Hover over element using Actions

```
public class HoverAction {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.get("https://example.com");
        WebElement element = driver.findElement(By.id("hoverTarget"));
        Actions actions = new Actions(driver);
        actions.moveToElement(element).perform();
        driver.quit();
    }
}
```

---

17. Drag and drop using Actions

```
public class DragAndDrop {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.get("https://example.com");
        WebElement source = driver.findElement(By.id("draggable"));
        WebElement target = driver.findElement(By.id("droppable"));
        Actions actions = new Actions(driver);
        actions.dragAndDrop(source, target).perform();
        driver.quit();
    }
}
```

---

18. Handle multiple windows

```
public class HandleMultipleWindows {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.get("https://example.com");
        String parent = driver.getWindowHandle();
        driver.findElement(By.id("openWindow")).click();
        Set<String> handles = driver.getWindowHandles();
        for (String handle : handles) {
            if (!handle.equals(parent)) {
                driver.switchTo().window(handle);
                break;
            }
        }
        driver.quit();
    }
```

```
}
```

---

19. Clear browser cookies and cache

```
public class ClearCookies {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.manage().deleteAllCookies();
        driver.get("https://example.com");
        driver.quit();
    }
}
```

---

20. Verify dropdown contains expected values

```
public class DropdownValidation {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.get("https://example.com");
        WebElement dropdown = driver.findElement(By.id("dropdown"));
        Select select = new Select(dropdown);
        List<String> expected = List.of("Option 1", "Option 2", "Option 3");
        List<String> actual = select.getOptions().stream().map(WebElement::getText).collect(Collectors.toList());
        System.out.println("Dropdown valid: " + actual.containsAll(expected));
        driver.quit();
    }
}
```

◆ API Testing / Backend Automation (Java + RestAssured)

---

1. Send GET request and validate status code

```
public class GetStatusCodeTest {
    public static void main(String[] args) {
        given()
            .baseUri("https://reqres.in/api")
        .when()
            .get("/users/2")
        .then()
            .statusCode(200)
            .log().all();
    }
}
```

---

2. Send POST request with JSON body from a file

```
public class PostWithJsonFile {
    public static void main(String[] args) {
        File file = new File("src/test/resources/user.json");

        given()
            .baseUri("https://reqres.in/api")
            .contentType("application/json")
            .body(file)
        .when()
            .post("/users")
        .then()
            .statusCode(201)
            .log().body();
    }
}
```

---

3. Chain two APIs (extract token, reuse in next call)

```
public class TokenChainingTest {
    public static void main(String[] args) {
        String token = given()
            .contentType("application/json")
            .body("{\"username\":\"admin\",\"password\":\"admin\"}")
        .when()
            .post("https://example.com/api/login")
```

```
        .then()
            .statusCode(200)
            .extract().path("token");

        given()
            .header("Authorization", "Bearer " + token)
        .when()
            .get("https://example.com/api/profile")
        .then()
            .statusCode(200);
    }
}
```

---

4. Validate fields inside a JSON response using JSONPath

```
public class JsonFieldValidation {
    public static void main(String[] args) {
        Response response = given()
            .get("https://reqres.in/api/users/2");

        JsonPath json = response.jsonPath();
        String email = json.getString("data.email");
        System.out.println("Email: " + email);
    }
}
```

---

5. Upload a file using multipart API request

```
public class FileUploadExample {
    public static void main(String[] args) {
        File file = new File("src/test/resources/sample.pdf");

        given()
            .multiPart("file", file)
            .post("https://example.com/api/upload")
        .then()
            .statusCode(200);
    }
}
```

---

6. Add headers and query params dynamically to request

```
public class DynamicHeaderQueryParam {
    public static void main(String[] args) {
        given()
            .header("Authorization", "Bearer myToken")
            .queryParam("page", 2)
        .when()
            .get("https://reqres.in/api/users")
        .then()
            .statusCode(200);
    }
}
```

---

7. Validate response time and schema of an API

```
public class ResponseTimeAndSchemaValidation {
    public static void main(String[] args) {
        given()
            .baseUri("https://reqres.in")
        .when()
            .get("/api/users/2")
        .then()
            .time(lessThan(2000L))
            .assertThat()
            .body(matchesJsonSchemaInClasspath("schema.json"));
    }
}
```

---

8. Create a reusable API utility class for GET/POST

```
public class ApiUtils {
```

```
    public static Response getRequest(String endpoint) {
        return given()
                .baseUri("https://reqres.in/api")
            .when()
                .get(endpoint);
    }

    public static Response postRequest(String endpoint, Object body) {
        return given()
                .baseUri("https://reqres.in/api")
                .contentType("application/json")
                .body(body)
            .when()
                .post(endpoint);
    }
}
```

---

9. Retry failed API call 3 times before failing

```
public class RetryLogicTest {
    public static void main(String[] args) {
        int maxRetries = 3;
        int count = 0;
        Response response = null;

        while (count < maxRetries) {
            response = given().get("https://reqres.in/api/users/2");
            if (response.statusCode() == 200) break;
            count++;
        }

        assert response != null;
        System.out.println("Final status: " + response.statusCode());
    }
}
```

---

10. Assert a nested JSON field using RestAssured

```
public class NestedJsonFieldAssert {
    public static void main(String[] args) {
        given()
            .baseUri("https://reqres.in")
        .when()
            .get("/api/users/2")
        .then()
            .body("data.first_name", equalTo("Janet"));
    }
}
```

---

11. RequestSpecification vs Response Parsing

```
public class RequestResponseParserDemo {
    public static void main(String[] args) {
        RequestSpecification reqSpec = given()
                .baseUri("https://api.example.com")
                .header("Authorization", "Bearer token")
                .contentType("application/json");

        Response response = reqSpec.when().get("/users/1");
        JsonPath json = response.jsonPath();
        System.out.println("Name: " + json.getString("name"));
    }
}
```

---

12. Response.asString() vs Response.as(Class)

```
public class ResponseParsingVariants {
    public static void main(String[] args) {
        Response res = given().get("https://api.example.com/users/1");

        String rawBody = res.asString();
```

```
        System.out.println("Raw Body: " + rawBody);

        User user = res.as(User.class);
        System.out.println("User Name from POJO: " + user.getName());
    }
}

class User {
    private int id;
    private String name;
    private String email;

    public String getName() { return name; }
    public String getEmail() { return email; }
    public int getId() { return id; }
}
```

---

13. Validate API with Path Parameters

```
public class PathParamTest {
    public static void main(String[] args) {
        given()
            .pathParam("id", 101)
        .when()
            .get("https://api.example.com/users/{id}")
        .then()
            .statusCode(200)
            .log().all();
    }
}
```

---

14. Validate API with Form Parameters

```
public class FormParamTest {
    public static void main(String[] args) {
        given()
            .contentType("application/x-www-form-urlencoded")
            .formParam("username", "testUser")
            .formParam("password", "pass123")
        .when()
            .post("https://api.example.com/login")
        .then()
            .statusCode(200);
    }
}
```

---

15. Handle Cookie in Request and Response

```
public class CookieExample {
    public static void main(String[] args) {
        Response response = given()
            .cookie("session_id", "xyz123")
        .when()
            .get("https://example.com/home");

        String cookie = response.getCookie("session_id");
        System.out.println("Session Cookie: " + cookie);
    }
}
```

---

16. Create and send a PATCH request

```
public class PatchExample {
    public static void main(String[] args) {
        given()
            .contentType("application/json")
            .body("{\"name\":\"updatedName\"}")
        .when()
            .patch("https://reqres.in/api/users/2")
        .then()
            .statusCode(200)
```

```
            .log().all();
    }
}
```

---

17. Delete request and validate response

```
public class DeleteExample {
    public static void main(String[] args) {
        given()
        .when()
            .delete("https://reqres.in/api/users/2")
        .then()
            .statusCode(204);
    }
}
```

---

18. Deserialize response array into POJO list

```
public class DeserializeArray {
    public static void main(String[] args) {
        Response response = given().get("https://reqres.in/api/users?page=2");
        List<User> users = response.jsonPath().getList("data", User.class);
        users.forEach(u -> System.out.println(u.getEmail()));
    }
}
```

---

19. Log request and response details

```
public class LoggingExample {
    public static void main(String[] args) {
        given()
            .log().all()
        .when()
            .get("https://reqres.in/api/users/2")
        .then()
            .log().all();
    }
}
```

---

20. Extract full response and use assertions

```
public class FullResponseExtract {
    public static void main(String[] args) {
        Response res = given().get("https://reqres.in/api/users/2");
        assertEquals(200, res.getStatusCode());
        assertTrue(res.asString().contains("Janet"));
    }
}
```