**Section 1: Selenium Introduction & Basics (Q1–Q20)**

1. **What is Selenium?**  
   An open-source suite for automating web browsers.
2. **Main components of Selenium?**  
   Selenium IDE, Selenium RC (deprecated), Selenium WebDriver, Selenium Grid.
3. **What is Selenium WebDriver?**  
   An API to control web browsers directly.
4. **Why use Selenium?**  
   Cross-browser testing, multi-language support, open-source.
5. **Programming languages supported by Selenium?**  
   Java, Python, C#, Ruby, JavaScript, Kotlin, etc.
6. **Browsers supported by Selenium?**  
   Chrome, Firefox, Edge, Safari, Opera, etc.
7. **Operating systems supported by Selenium?**  
   Windows, macOS, Linux, Unix.
8. **What is Selenium IDE?**  
   A browser extension for record & playback testing.
9. **What is Selenium Grid?**  
   Tool for parallel test execution across machines/browsers.
10. **Difference between Selenium WebDriver and IDE?**  
    WebDriver – code-based, flexible; IDE – record & playback, limited.
11. **Limitation of Selenium?**  
    Cannot automate desktop applications.
12. **Default browser for Selenium execution?**  
    Older versions defaulted to Firefox.
13. **Basic architecture of WebDriver?**  
    Test Script → Language Bindings → JSON Wire Protocol → Browser Driver → Browser.
14. **What is JSON Wire Protocol?**  
    Communication protocol between WebDriver and browser.
15. **What is a browser driver?**  
    Middle layer that executes WebDriver commands in the browser.
16. **What is ChromeDriver?**  
    Executable for Chrome browser automation.
17. **What is GeckoDriver?**  
    Executable for Firefox browser automation.
18. **What is WebDriver Manager?**  
    A library to manage browser drivers automatically.
19. **When to use Selenium Standalone Server?**  
    For Selenium Grid or remote execution.
20. **Selenium vs QTP/UFT?**  
    Selenium is open-source and web-only; QTP/UFT is paid and supports web/desktop.

**Section 2: Locators in Selenium (Q21–Q40)**

1. **What is a locator?**  
   A way to identify web elements.
2. **Types of locators in Selenium?**  
   ID, Name, Class Name, Tag Name, Link Text, Partial Link Text, CSS Selector, XPath.
3. **Fastest locator?**  
   ID (if available).
4. **When to use Name locator?**  
   When there’s a unique name attribute.
5. **Class Name locator limitation?**  
   Multiple classes may cause errors.
6. **When to use Tag Name locator?**  
   To find elements of the same type.
7. **When to use Link Text locator?**  
   To locate links with exact visible text.
8. **When to use Partial Link Text?**  
   When link text is long, to partially match.
9. **What is a CSS Selector?**  
   Uses CSS syntax to locate elements.
10. **CSS Selector vs XPath?**  
    CSS is faster; XPath is more flexible.
11. **How to use ID in CSS Selector?**  
    #idValue
12. **How to use Class in CSS Selector?**  
    .classValue
13. **How to use attributes in CSS Selector?**  
    [attribute='value']
14. **What is XPath?**  
    XML Path Language to traverse DOM and locate elements.
15. **What is Absolute XPath?**  
    Starts from root (/html/body/...).
16. **What is Relative XPath?**  
    Starts from current node (//tagname[@attr='value']).
17. **How to use OR in XPath?**  
    //tag[@attr='value1' or @attr='value2']
18. **How to use AND in XPath?**  
    //tag[@attr1='val1' and @attr2='val2']
19. **How to use text() in XPath?**  
    //tag[text()='value']
20. **How to use contains() in XPath?**  
    //tag[contains(text(),'partialText')]

**Section 3: WebElement Methods (Q41–Q60)**

1. **What is a WebElement?**  
   Represents an HTML element in Selenium.
2. **What does click() do?**  
   Clicks the element.
3. **What does sendKeys() do?**  
   Types text into an input field.
4. **What does clear() do?**  
   Clears value in an input field.
5. **What does getText() do?**  
   Returns visible text of an element.
6. **What does getAttribute() do?**  
   Returns the value of an HTML attribute.
7. **What does isDisplayed() do?**  
   Checks if element is visible.
8. **What does isEnabled() do?**  
   Checks if element is enabled.
9. **What does isSelected() do?**  
   Checks if checkbox/radio is selected.
10. **What does submit() do?**  
    Submits a form.
11. **What does getCssValue() do?**  
    Returns CSS property value.
12. **What does getTagName() do?**  
    Returns HTML tag name.
13. **What does findElement() do?**  
    Locates a single element.
14. **What does findElements() do?**  
    Locates multiple elements.
15. **What does getSize() do?**  
    Returns element height and width.
16. **What does getLocation() do?**  
    Returns element’s (X,Y) position.
17. **What does getRect() do?**  
    Returns element’s size and location.
18. **What does getScreenshotAs() do?**  
    Captures screenshot of the element.
19. **What does clickAndHold() do?**  
    Holds down the mouse button.
20. **What does release() do?**  
    Releases the mouse button.

**Section 4: Waits & Browser Handling (Q61–Q80)**

1. **What is an implicit wait?**  
   Default wait time for locating elements.
2. **What is an explicit wait?**  
   Wait for a specific condition.
3. **What is a fluent wait?**  
   Wait with polling frequency.
4. **Can Thread.sleep() be used?**  
   Yes, but not recommended.
5. **What is WebDriverWait?**  
   Class for implementing explicit waits.
6. **What is ExpectedConditions?**  
   Class with common wait conditions.
7. **What does titleIs() do?**  
   Checks for exact page title.
8. **What does urlContains() do?**  
   Checks if URL contains specific text.
9. **What does elementToBeClickable() do?**  
   Waits until element is visible and enabled.
10. **What does alertIsPresent() do?**  
    Waits until alert appears.
11. **How to maximize browser window?**  
    driver.manage().window().maximize();
12. **What is a window handle?**  
    Unique ID for a browser window.
13. **What does getWindowHandle() do?**  
    Returns current window’s handle.
14. **What does getWindowHandles() do?**  
    Returns handles of all open windows.
15. **What does switchTo().window() do?**  
    Switches to another window.
16. **What does switchTo().frame() do?**  
    Switches to an iframe.
17. **What does switchTo().defaultContent() do?**  
    Switches back to main document.
18. **What does switchTo().parentFrame() do?**  
    Switches to parent frame.
19. **What does switchTo().alert() do?**  
    Switches to alert.
20. **What does alert.accept() do?**  
    Clicks OK on alert.

**Section 5: Common Automation Tasks (Q81–Q100)**

1. **What does alert.dismiss() do?**  
   Clicks Cancel/No in an alert.
2. **What does alert.getText() do?**  
   Retrieves the alert message text.
3. **What does alert.sendKeys() do?**  
   Sends input text to a prompt alert.
4. **What is the Select class?**  
   A class to handle dropdown elements.
5. **What does selectByVisibleText() do?**  
   Selects dropdown option by visible text.
6. **What does selectByValue() do?**  
   Selects dropdown option by value attribute.
7. **What does selectByIndex() do?**  
   Selects dropdown option by index number.
8. **What does getOptions() do (Select class)?**  
   Returns all options in a dropdown.
9. **What does getFirstSelectedOption() do?**  
   Returns the first selected dropdown option.
10. **What does deselectAll() do?**  
    Deselects all options in a multi-select dropdown.
11. **Why use the Robot class in Selenium?**  
    For keyboard/mouse actions outside WebDriver’s scope.
12. **What is TakesScreenshot?**  
    An interface for capturing screenshots.
13. **What does getScreenshotAs() (driver) do?**  
    Captures a screenshot of the browser.
14. **How to upload a file in Selenium?**  
    Use sendKeys() with the file path.
15. **How to perform drag and drop?**  
    Use Actions class dragAndDrop() method.
16. **How to perform mouse hover?**  
    Use Actions class moveToElement().
17. **How to perform double click?**  
    Use Actions class doubleClick().
18. **How to perform right click?**  
    Use Actions class contextClick().
19. **How to scroll a page?**  
    Use JavaScriptExecutor executeScript().
20. **What is JavaScriptExecutor?**  
    An interface to execute JavaScript in the browser.

**Section 6: Page Object Model (POM) & Framework Basics (Q101–Q120)**

1. **What is Page Object Model (POM)?**  
   A design pattern where each page has a separate class representing its elements and actions.
2. **Why use POM?**  
   Improves code reusability, maintainability, and readability.
3. **What is a Page Class in POM?**  
   Contains locators and methods for one page.
4. **What is PageFactory?**  
   A helper class for initializing elements with annotations.
5. **What is @FindBy?**  
   An annotation to define element locators.
6. **What does PageFactory.initElements() do?**  
   Initializes elements in a Page class.
7. **POM vs PageFactory?**  
   POM is the concept; PageFactory is an implementation helper.
8. **How does SRP apply in POM?**  
   Each page class handles only that page’s elements/actions.
9. **How to use a Page Object in a Test Class?**  
   Create an instance and call its methods.
10. **Why use a constructor in POM?**  
    To initialize driver and elements.
11. **Downside of scripts without POM?**  
    Duplicate code, harder maintenance.
12. **Why use POM with TestNG/JUnit?**  
    Combines structured design with execution control.
13. **How to make reusable methods in a framework?**  
    Put them in utility classes.
14. **What is a Base Class in a framework?**  
    Holds common setup, teardown, and configurations.
15. **What is a config properties file?**  
    Stores external configuration values.
16. **How to load config data?**  
    Use Java Properties class.
17. **How to do data-driven testing in POM?**  
    Read from Excel, CSV, JSON, or a database.
18. **Why create a constants class?**  
    Centralize fixed values.
19. **Typical framework folder structure?**  
    Pages, Tests, Utils, Config, Reports, Drivers.
20. **How easy is POM maintenance?**  
    Change in UI affects only its page class.

**Section 7: TestNG & JUnit Integration (Q121–Q140)**

1. **What is TestNG?**  
   A testing framework with advanced features like grouping, parallel execution.
2. **TestNG vs JUnit?**  
   TestNG has more advanced features; JUnit is simpler.
3. **What is @Test in TestNG?**  
   Marks a method as a test method.
4. **What is @BeforeMethod?**  
   Runs before each test method.
5. **What is @AfterMethod?**  
   Runs after each test method.
6. **What is @BeforeClass?**  
   Runs once before the class’s tests.
7. **What is @AfterClass?**  
   Runs once after the class’s tests.
8. **What is @BeforeTest?**  
   Runs before <test> in XML.
9. **What is @AfterTest?**  
   Runs after <test> in XML.
10. **What is @BeforeSuite?**  
    Runs once before the suite.
11. **What is @AfterSuite?**  
    Runs once after the suite.
12. **What is testng.xml?**  
    Configuration file for TestNG.
13. **What are groups in TestNG?**  
    Categories for grouping tests.
14. **How to set dependencies in TestNG?**  
    Use dependsOnMethods or dependsOnGroups.
15. **What is DataProvider in TestNG?**  
    Supplies multiple sets of data to a test.
16. **DataProvider return type?**  
    Object[][]
17. **How to pass parameters in TestNG?**  
    Define in XML or use @Parameters.
18. **What is a Retry Analyzer?**  
    Mechanism to rerun failed tests.
19. **How to enable parallel execution in TestNG?**  
    Set parallel attribute in XML.
20. **How to set priority in TestNG?**  
    @Test(priority = 1)

**Section 8: Selenium Grid & Parallel Execution (Q141–Q160)**

1. **What is Selenium Grid?**  
   Runs tests in parallel on multiple machines/browsers.
2. **What is a Hub in Grid?**  
   Central server managing tests.
3. **What is a Node in Grid?**  
   Machine connected to hub running tests.
4. **Hub and Node communication?**  
   Via JSON Wire Protocol.
5. **Types of Grid setups?**  
   Standalone, Hub-Node, Fully distributed.
6. **How to do cross-browser testing in Grid?**  
   Set browser name/version in capabilities.
7. **How to set parallel execution in Grid?**  
   Configure in TestNG or code.
8. **What is Docker-Selenium?**  
   Running Grid in Docker containers.
9. **How to run Chrome/Firefox nodes in Docker?**  
   Use docker-compose.
10. **Advantages of Grid?**  
    Saves time, supports multiple environments.
11. **What is version mismatch in Grid?**  
    Driver and browser version mismatch.
12. **What is RemoteWebDriver?**  
    Class to run tests remotely.
13. **Constructor parameters for RemoteWebDriver?**  
    Hub URL and capabilities.
14. **Capabilities vs Options?**  
    Capabilities are old; Options are browser-specific.
15. **What is BrowserStack/Sauce Labs?**  
    Cloud Selenium providers.
16. **Advantages of Cloud Grid?**  
    Real devices/browsers, no setup needed.
17. **What is session timeout in Grid?**  
    Time after which idle sessions close.
18. **What is node timeout in Grid?**  
    Time before idle node disconnects.
19. **What is maxSession in Grid?**  
    Max tests a node can run at once.
20. **Risk in parallel execution?**  
    Data conflicts from shared resources.

**Section 9: Reporting & Logging (Q161–Q180)**

1. **What is Extent Reports?**  
   A library for generating detailed HTML reports.
2. **How to integrate Extent Reports in Selenium?**  
   Create ExtentReports and ExtentTest instances, log steps, and flush.
3. **What does ExtentTest.log() do?**  
   Adds log entries for each step.
4. **What does ExtentReports.flush() do?**  
   Writes all log data to the report file.
5. **How to add screenshots in a report?**  
   Capture with getScreenshotAs() and attach to the report.
6. **What is Allure Report?**  
   A lightweight, detailed reporting framework.
7. **How to integrate Allure with Selenium?**  
   Use TestNG listeners or annotations.
8. **What is Log4j?**  
   A Java logging framework.
9. **What is a Log4j config file?**  
   Defines logging levels, appenders, and patterns.
10. **Logging levels in Log4j?**  
    TRACE, DEBUG, INFO, WARN, ERROR, FATAL.
11. **Why use logging in automation?**  
    To debug and trace execution flow.
12. **What is SLF4J?**  
    A logging abstraction layer.
13. **Difference between reporting and logging?**  
    Reports summarize execution; logs detail each step.
14. **How to create a custom HTML report?**  
    Implement TestNG’s Reporter or IReporter.
15. **What is a listener in TestNG?**  
    Interface to capture test events.
16. **What is onTestSuccess() in ITestListener?**  
    Executes when a test passes.
17. **What is onTestFailure() in ITestListener?**  
    Executes when a test fails.
18. **What does Reporter.log() do?**  
    Adds messages to TestNG default report.
19. **How to set custom screenshot names?**  
    Combine test method name with timestamp.
20. **How to integrate reports in CI/CD?**  
    Publish report artifacts in build pipelines.

**Section 10: Best Practices & Troubleshooting (Q181–Q200)**

1. **How to improve test stability in Selenium?**  
   Use explicit waits instead of hard waits.
2. **Why does ElementNotVisibleException occur?**  
   Element exists in DOM but is not visible.
3. **Why does NoSuchElementException occur?**  
   Wrong locator or element not present.
4. **Why does StaleElementReferenceException occur?**  
   DOM refreshed, old reference is invalid.
5. **Why does ElementClickInterceptedException occur?**  
   Another element covers the target element.
6. **Why does TimeoutException occur?**  
   Wait condition not met within time.
7. **Why does WebDriverException occur?**  
   Communication issue between driver and browser.
8. **How to avoid data conflicts in parallel execution?**  
   Use ThreadLocal WebDriver instances.
9. **How to set page load timeout?**  
   driver.manage().timeouts().pageLoadTimeout().
10. **How to set script timeout?**  
    driver.manage().timeouts().setScriptTimeout().
11. **How to clear browser cache in tests?**  
    Delete cookies or use incognito mode.
12. **How to maximize browser in every test?**  
    Put maximize code in Base class setup.
13. **What is headless browser mode?**  
    Browser runs without a GUI.
14. **Why use headless mode?**  
    Faster execution and suitable for CI/CD.
15. **What is ChromeOptions?**  
    Class to customize Chrome browser settings.
16. **What is FirefoxOptions?**  
    Class to customize Firefox browser settings.
17. **How to load browser extensions in tests?**  
    Add extension path in Options.
18. **How to set proxy in Selenium?**  
    Create a Proxy object and add to capabilities.
19. **Difference between driver.quit() and driver.close()?**  
    quit() closes all windows; close() closes current window.
20. **Selenium automation best practices?**  
    Use POM, explicit waits, reusable methods, externalized data, logs/reports.

**Section 11: Selenium Configuration & Environment (Q201–Q220)**

1. **What is Selenium Standalone JAR?**  
   JAR for running Selenium Server/Grid.
2. **What are Selenium language bindings?**  
   Language-specific APIs for Selenium.
3. **How to set driver executable path?**  
   System.setProperty("webdriver.chrome.driver", "path");
4. **Why use WebDriver Manager?**  
   Auto-manages driver versions.
5. **Why use Maven for Selenium projects?**  
   Manage dependencies and builds.
6. **What is pom.xml?**  
   Maven configuration file for dependencies.
7. **How to add Selenium Maven dependency?**  
   Add coordinates in <dependency> tag.
8. **How to add TestNG in Maven?**  
   Add TestNG dependency in pom.xml.
9. **How to check Selenium’s latest version?**  
   Maven Central Repository or official site.
10. **How to run Selenium without setting driver path?**  
    Use WebDriver Manager or set in PATH.
11. **How to add driver path to PATH variable?**  
    Add driver folder path in OS environment variables.
12. **What does Maven clean do?**  
    Deletes the target folder.
13. **What does Maven test do?**  
    Runs tests.
14. **Difference between src/test/java and src/main/java?**  
    src/main/java for app code; src/test/java for tests.
15. **How to make timeouts configurable?**  
    Store in properties file or constants.
16. **Why externalize test data?**  
    Change data without touching code.
17. **How to read Excel test data?**  
    Use Apache POI or JExcel API.
18. **How to read JSON test data?**  
    Use JSON Simple or Jackson.
19. **How to read properties file?**  
    Use Java Properties class.
20. **How to run tests on different browsers?**  
    Pass browser name as parameter from config/XML.

**Section 12: Common WebDriver Commands (Q221–Q240)**

1. **driver.get() vs driver.navigate().to()?**  
   get() is simpler; navigate().to() maintains history.
2. **What does navigate().back() do?**  
   Goes to the previous page.
3. **What does navigate().forward() do?**  
   Goes to the next page.
4. **What does navigate().refresh() do?**  
   Reloads the current page.
5. **What does getTitle() do?**  
   Returns page title.
6. **What does getCurrentUrl() do?**  
   Returns current URL.
7. **What does getPageSource() do?**  
   Returns HTML source code.
8. **What does setSize() do?**  
   Changes browser window size.
9. **What does setPosition() do?**  
   Changes browser window position.
10. **What does deleteAllCookies() do?**  
    Deletes all cookies.
11. **What does addCookie() do?**  
    Adds a cookie.
12. **What does getCookies() do?**  
    Returns all cookies.
13. **What does implicitlyWait() do?**  
    Sets default implicit wait time.
14. **What does pageLoadTimeout() do?**  
    Sets max page load time.
15. **What does setScriptTimeout() do?**  
    Sets max time for async scripts.
16. **What does close() do?**  
    Closes current browser window.
17. **What does quit() do?**  
    Closes all browser windows.
18. **What does switchTo().window() do?**  
    Switches to another window.
19. **What does switchTo().frame() do?**  
    Switches to a frame.
20. **What does switchTo().defaultContent() do?**  
    Switches to the main document.

**Section 13: Element Handling & Actions (Q241–Q260)**

1. **When does isDisplayed() return false?**  
   If element is hidden or CSS display is none.
2. **When does isEnabled() return false?**  
   If element has disabled attribute.
3. **When does isSelected() return true?**  
   If checkbox/radio is checked.
4. **Why use Actions class?**  
   For advanced mouse and keyboard actions.
5. **What does moveToElement() do?**  
   Moves mouse to element.
6. **What does clickAndHold() do?**  
   Holds mouse button.
7. **What does release() do?**  
   Releases mouse button.
8. **What does contextClick() do?**  
   Performs right click.
9. **What does doubleClick() do?**  
   Performs double click.
10. **What does dragAndDrop() do?**  
    Drags and drops element.
11. **What is Keys class?**  
    Contains keyboard key constants.
12. **What does sendKeys(Keys.TAB) do?**  
    Presses Tab key.
13. **How to initialize Robot class?**  
    Robot robot = new Robot();
14. **What does robot.keyPress() do?**  
    Presses a key.
15. **What does robot.keyRelease() do?**  
    Releases a key.
16. **What does isMultiple() in Select do?**  
    Checks if dropdown allows multiple selection.
17. **What does deselectByValue() do?**  
    Deselects option by value.
18. **What does deselectByIndex() do?**  
    Deselects option by index.
19. **What does deselectByVisibleText() do?**  
    Deselects option by visible text.
20. **What does getAllSelectedOptions() do?**  
    Returns all selected options.

**Section 14: Practical Scenarios (Q261–Q280)**

1. **How to handle dynamic elements?**  
   Use XPath contains() or starts-with().
2. **How to click hidden elements?**  
   Use JavaScriptExecutor.
3. **How to send text to disabled element?**  
   Use JavaScriptExecutor.
4. **How to find broken links?**  
   Check HTTP status of each anchor href.
5. **How to check if an image is loaded?**  
   Use JS to check naturalWidth property.
6. **How to find window order when handling multiple windows?**  
   Iterate through window handles.
7. **How to read table data?**  
   Loop through table rows and columns.
8. **How to search in paginated table?**  
   Loop pages and search.
9. **How to verify file download?**  
   Check if file exists in download folder.
10. **How to verify file upload?**  
    Check success message or uploaded file name.
11. **How to handle auto-suggestion dropdown?**  
    Fetch list and click desired option.
12. **How to handle date picker?**  
    Navigate month/year and click date.
13. **How to handle slider?**  
    Use dragAndDropBy().
14. **How to fetch tooltip text?**  
    Get attribute "title" or hover then getText().
15. **How to handle modal dialog?**  
    Verify text and click close.
16. **How to scrape infinite scroll page?**  
    Loop scroll until no new data.
17. **How to add session cookie?**  
    Use driver.manage().addCookie().
18. **How to delete a session cookie?**  
    Use driver.manage().deleteCookieNamed().
19. **How to enable mobile emulation?**  
    Set device metrics in ChromeOptions.
20. **How to set network conditions in automation?**  
    Use Chrome DevTools Protocol.

**Section 15: Integration, CI/CD & Troubleshooting (Q281–Q300)**

1. **How to run Selenium tests in Jenkins?**  
   Add Maven build step.
2. **How to show reports in Jenkins?**  
   Use HTML Publisher Plugin.
3. **How to manage Selenium project in Git?**  
   Push source code to repository.
4. **How to run tests on pull requests?**  
   Add test stage in CI pipeline.
5. **How to run Selenium in Docker?**  
   Use docker-compose or Selenium images.
6. **Examples of cloud Selenium providers?**  
   BrowserStack, Sauce Labs, LambdaTest.
7. **Advantages of cloud testing?**  
   Real browsers/devices, no local setup.
8. **Why parallel execution in CI?**  
   Reduce execution time.
9. **How to do cross-browser testing in CI?**  
   Add different browser configs in pipeline.
10. **Common Selenium CI failures?**  
    Element not found, timeouts, version mismatches.
11. **How to fix driver version mismatch?**  
    Update driver or downgrade browser.
12. **Why use headless mode in CI?**  
    Faster execution, no GUI.
13. **Why does element overlapping happen?**  
    Layout issues or dynamic popups.
14. **Risk of mixing implicit and explicit waits?**  
    Unexpected delays and timeouts.
15. **What is a flaky test?**  
    Test that passes/fails inconsistently.
16. **How to fix flaky tests?**  
    Stable waits, proper sync, data isolation.
17. **Why use retry logic in CI?**  
    To rerun temporary failures.
18. **How to enable video recording in CI?**  
    Use cloud provider or recording library.
19. **How to save failure screenshots in CI?**  
    Add screenshot code in onTestFailure().
20. **Best CI/CD practice for Selenium tests?**  
    Parallel runs, waits, version control, reporting, retries.