1. CONSIDER IN A WEBPAGE THERE ARE 3 WEBELEMENTS WITH SAME ID VALUE SAY ID='PRODUCT' WHAT WILL FINDELEMENT DO IN THIS CASE? WHICH ID WILL GET LOCATED?

The first id value will be located here because findElement always clicks/performs actions in the first matching web element because findElement

will return the first matching criteria.

If you want to perform actions on the third web element, we can use indexing and locate the particular web element.

2. HOW MANY TEST CASES YOU AUTOMATE PER SPRINT?

In a two week sprint, we automate around

23 critical test cases.

56 major test cases.

1012 minor test cases.

3 IS IT POSSIBLE TO AUTOMATE EVERYTHING? 100% AUTOMATION POSSIBLE OR NOT?

No, it's not possible to automate everything

There are few scenarios where automation is not possible. We depend on manual testing there.

4 WHAT ARE THE CHALLENGES YOU HAVE FACED AS AN AUTOMATION ENGINEER?

Automating applications with captcha was difficult.

Handling dynamic web elements was difficult.

Integrating framework with CI/CD pipeline was a major challenge we overcame.

5 HOW DO YOU PERFORM CODE OPTIMIZATION?

Perform regular code refactoring.

Remove sleep commands.

Avoid using implicit wait and use explicit wait.

Avoid code duplication.

5. HOW DO YOU INTERACT WITH DYNAMIC WEB ELEMENTS? WHAT TECHNIQUES YOU USED TO LOCATE DYNAMIC WEB ELEMENTS.

Suppose when there are no direct ways to locate web elements, we can go for absolute XPath strategy.

From root node to desired web element we can do parentchild traversal and locate web elements.

Traverse from top of the HTML tree to desired web element /form/div[1]/div/div[1]/div/div/input[1].

6. WHY DO WE HAVE TO USE WAIT MECHANISM IN RUNNING SELENIUM TESTS?

Sometimes it takes longer time for a web element to load/appear.

At that time our tests might throw NoSuchElementException; to handle this case we use Selenium waits like implicit and explicit wait.

Wait commands help to ensure that the automation test script waits for certain conditions to be met before executing scripts.

Ensure the stability, reliability, and accuracy of test scripts.

7. HOW DOES PARALLEL EXECUTION BENEFIT YOU?

Running our tests on multiple platforms/nodes means splitting our tests i.e., it can save a large amount of

test execution time by performing parallel testing.

Consider our project has 1,000 test cases; through parallel execution we can split our tests and run 500 test cases on one device and 500 on another device.

We can achieve parallel execution by using Selenium Grid, TestNG.

8. How will you handle dynamic dropdowns in Selenium?

First use wait techniques to make sure the dropdown is loaded.

Once dropdown is visible, locate the dropdown using any locator (id, xpath, etc.).

Select values from the dropdown using methods like selectByValue(), selectByVisibleText(), or selectByIndex().

Dynamic dropdown = allows a second dropdown field to display values based on the selection made in the first dropdown field.

9. What is the main usage of TestNG Library?

The major uses of TestNG are:

TestNG generates reports automatically (Passed/Failed test case counts will be present).

TestNG annotations make the code more readable.

TestNG supports parallel execution.

TestNG supports grouping test cases.

10. Can we use Selenium for mobile testing?

With Selenium alone, we cannot do automated mobile testing.

But we can combine Selenium with Appium, an open-source mobile test automation framework, to perform

automated mobile testing.

11. What are the components of your framework?

Baseclass

Resources package

Object repository

Tests package

Reporting

Dockerfile, YAML file for CI/CD execution

pom.xml

testng.xml

12. Explain your automation process.

Review requirements Create test script Send it for review After approval, create PR and merge it to master branch.

13. Do you prefer soft assertion or hard assertion?

Soft assertion is more preferred because with hard assertions, if the actual result does not match the expected,

the assert statement fails and the test haltsthe remaining tests are skipped and marked as failed.

In contrast, with soft assertions, the verification statement may fail but the test continues, and the failure is recorded in the test results.

14. What is the use of Robot class?

Robot class is a Java-based utility.

It lets the tester automate tasks that cant be done using Seleniums built-in methodssuch as simulating keyboard and mouse interactions.

15. How will you validate text color or CSS attributes using Selenium?

Use getCssValue() to get the color of text/button, background color, etc.

getCssValue() returns a value in RGBA format like (255,255,255,1).

We can convert RGBA value to hex using Color.fromString().asHex() #ffffff.

16. How do you confirm the exact position of a web element in the webpage?

Use verifyElementPositionLeft and verifyElementPositionTop.

They use pixel comparison to determine the position of the element from the left and top of the page.

17. What is headless driver?

Headless drivers are used for faster test execution.

Example: launching applications in Chrome headless drivers runs tests faster compared to normal UI-based browser execution.

18. How does your Selenium test script interact with browsers? How is the communication channel established?

A: Selenium uses the JSON Wire Protocol to communicate between your test script and the browser. This is

a RESTful web service mechanism using JSON over HTTP.

19. What is your preferred locator for identifying web elements?

A: Selenium offers many locatorsid, xpath, cssSelector, name, className, linkText, etc. The preferred approach:

Use id if availableit's unique and fast.

If no id, use xpath or cssSelector.

20. When do you use absolute XPath strategy? Handling dynamic web elements?

A: You opt for absolute XPath only when no direct locators exist.

It involves traversing from the root node down to the desired element via parent-child paths.

21. Do you prefer absolute XPath or relative XPath?

A: Relative XPath is preferred because it provides a more direct way to locate elements.

Absolute XPath is brittleHTML structure changes can break it.

22. What is the purpose of using waits in Selenium?

A: Waits help manage delays in loading or rendering elements. Without waits, tests might throw NoSuchElementException.

Selenium supports both implicit and explicit waits for this purpose.

23. Explain the different types of waits used in Selenium.

Implicit wait: Set globally; defines a default waiting time throughout the test framework.

Explicit wait: Used for specific scenarios, like waiting for visibility of an element or for an alert.

Fluent wait: Customizable wait with polling intervals, useful for elements that take varying time to appear.

24. What is the use of assertion in Selenium?

A: Assertions verify whether actual results match expected outcomese.g.,

checking page titles, presence, visibility, selection, or enabled state of elements using Assert.assertEquals(...).

25. What design pattern do you follow for test scripting?

A: The Page Object Model (POM). This pattern creates page-specific object repository classes that locate web elements and encapsulate actions on those pages.

It enhances code reusability and readability.

26. Write an XPath for a link that contains text Im in a meeting.

A: Use a contains-based XPath:

//tagname[contains(text(), 'meeting')]

27. How to mouse hover over a web element?

A: Use Seleniums Actions class. Example:

Actions action = new Actions(driver);

action.moveToElement(driver.findElement(By.id("element-id"))).perform();

28. What is Page Factory?

A: Page Factory is an extension of the Page Object Model (POM).

It is used to initialize web elements with @FindBy annotations, making the code cleaner and more readable.

Example:

@FindBy(id = "username") WebElement username;

PageFactory.initElements(driver, this);

29.How do you implement TestNG framework in your project?

TestNG is used for managing test cases, grouping, prioritizing, and generating reports.

Create a class with @Test methods.

Use @BeforeSuite, @BeforeTest, @BeforeMethod annotations for setup.

Execute with testng.xml.

30. What are the annotations used in TestNG?

A: Some key TestNG annotations:

@BeforeSuite

@BeforeTest

@BeforeClass

@BeforeMethod

@Test

@AfterMethod

@AfterClass

@AfterTest

@AfterSuite

31. How do you run only failed test cases in TestNG?

A: After executing tests, TestNG creates a test-output folder with a testng-failed.xml file.

You can run this XML file to re-execute only the failed test cases.

32. What is the difference between driver.close() and driver.quit()?

A:

driver.close(): Closes the current browser window only.

driver.quit(): Quits the entire browser session, closing all windows and ending the WebDriver session.

33. How to handle dropdown in Selenium?

A: Use the Select class in Selenium:

```
Select select = new Select(driver.findElement(By.id("dropdown")));
select.selectByVisibleText("Option");
select.selectByValue("value");
select.selectByIndex(2);
```

34. How do you launch browser in a particular dimension / What is the use of ChromeOptions?

We can use ChromeOptions class to customize and configure a ChromeDriver session.

ChromeOptions helps in performing operations like:

Opening Chrome in headless mode.

Setting browser size.

Launching browser in maximized mode.

```
if (browser.equalsIgnoreCase("chrome")) {
    ChromeOptions opts = new ChromeOptions();
    opts.addArguments("--start-maximized");
    opts.addArguments("--window-position=1920,1080");
}
```

35. What is the use of DesiredCapabilities?

Used to set the browser properties.

Properties include:

Browser name

Settings

Version

Platform

```
DesiredCapabilities capability = DesiredCapabilities.chrome();
capability.setBrowserName("chrome");
capability.setPlatform(Platform.WIN8_1);
driver = new RemoteWebDriver(new URL(nodeUrl), capability);
```

36. Why is WebDriver called an Interface?

WebDriver is a public interface because it defines a set of methods.

The interface mechanism achieves abstraction (abstract methods have no body).

Interfaces define methods for other classes to implement.

The implementation is provided by browser-specific classes.

Examples:

ChromeDriver, FirefoxDriver, SafariDriver, etc.

37. How do you verify whether a webpage contains 404 error code during page launch?

Use getPageSource() to check if page contains any "404" errors during launch.

Assert.assertTrue(driver.getPageSource().contains("404"));

38. In which scenario have you used findElements?

findElements is a method used to find a list of web elements on a webpage.

Explanation:

findElements Returns a list of web elements.

Practical scenario: Used findElements to fetch all elements on a webpage during a case/test.

1. What is Selenium? What are its components?

Selenium is an open-source automation tool for web applications.

Components:

Selenium WebDriver

Selenium IDE

Selenium Grid

Selenium RC (deprecated)

2. What is Selenium WebDriver?

Its a programming interface to automate browser actions like clicking, typing, validating, etc. It directly communicates with the browser without a separate server.

3. Difference between findElement() and findElements()?

Method Description

findElement() Returns single WebElement, throws exception if not found

findElements() Returns list of WebElements, returns empty list if none found

4. How do you handle dropdowns in Selenium?

Using the Select class:

```
Select dropdown = new Select(driver.findElement(By.id("dropdownId")));
dropdown.selectByVisibleText("Option");
```

5. How do you handle alerts in Selenium?

```
Alert alert = driver.switchTo().alert();
alert.accept(); // or alert.dismiss();
```

6. What are different waits in Selenium?

Implicit Wait waits globally

Explicit Wait waits for a specific condition

```
WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("element")));
```

7. How do you handle multiple windows

```
String parent = driver.getWindowHandle();
Set<String> allWindows = driver.getWindowHandles();
for (String child : allWindows) {
    if (!child.equals(parent)) {
        driver.switchTo().window(child);
    }
}
```

8. How do you handle frames/iframes?

```
driver.switchTo().frame("frameName"); // or by index or WebElement
// do actions inside frame
```

driver.switchTo().defaultContent(); // to come back

9. How do you capture screenshots in Selenium?

File src = ((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);

FileUtils.copyFile(src, new File("path/to/save.png"));

10. How do you validate text or elements in Selenium?

String actual = driver.findElement(By.id("msg")).getText();

Assert.assertEquals(actual, "Expected Message");

11. What are the different locators in Selenium?

id

name

className

tagName

linkText

partialLinkText

cssSelector

xpath

12. What is the difference between XPath and CSS Selector?

XPath CSS Selector

Can traverse both forward and backward Only forward

Slower in performance Faster

Syntax is complex Simpler

13. How do you perform mouse hover and keyboard actions?

Using Actions class:

Actions act = new Actions(driver);

act.moveToElement(element).click().build().perform();

14. What is POM (Page Object Model)?

POM is a design pattern that creates an object repository for web elements. It improves:

Code reusability

Maintainability

Readability

15. What is TestNG? Why do we use it with Selenium?

TestNG is a testing framework (like JUnit) used for:

Test case management

Reporting

Parallel execution

Grouping and prioritizing tests

16. What is the use of @BeforeMethod, @Test, @AfterMethod in TestNG?

@BeforeMethod: runs before each test

@Test: marks a test case

@AfterMethod: runs after each test

17. How do you run tests in multiple browsers?

By setting WebDriver dynamically:

```
if (browser.equalsIgnoreCase("chrome")) {
    driver = new ChromeDriver();
} else if (browser.equalsIgnoreCase("firefox")) {
    driver = new FirefoxDriver();
}
```

18. How do you manage waits and avoid ElementNotInteractable exceptions?

Use explicit waits, avoid hard sleeps, wait for visibility or clickability.

19. How do you perform data-driven testing in Selenium?

Using TestNG @DataProvider

Reading data from Excel via Apache POI

20. How do you integrate Selenium with Maven or Jenkins?

Maven: For dependency management via pom.xml

Jenkins: For CI/CD pipeline run Selenium test suite on each code push