[Explore Software Testing](https://explore2test.blogspot.com/)

Top of Form

Bottom of Form

AT\_Assignmnets\_Q&A

*on*[*January 15, 2025*](https://explore2test.blogspot.com/2025/01/atassignmnetsq.html)

**1. What is Selenium, and how does it help in web testing?**

**Answer**:  
Selenium is an open-source tool used to automate web browsers. It allows you to write code (in Java, Python, etc.) to control a web browser (like Chrome or Firefox) and simulate real user actions, such as clicking buttons, filling forms, and checking if things are displayed correctly on a webpage. This makes web testing easier and faster.

**2. Write a simple Selenium script to open a website and verify the page title.**

**Answer**: Here's a basic Selenium script in Java to open a website (like Google) and verify the page title:

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

public class VerifyTitle {

public static void main(String[] args) {

// Set the path to the ChromeDriver executable

System.setProperty("webdriver.chrome.driver", "path/to/chromedriver");

// Create an instance of ChromeDriver

WebDriver driver = new ChromeDriver();

// Open Google homepage

driver.get("https://www.google.com");

// Verify the page title

String pageTitle = driver.getTitle();

if(pageTitle.equals("Google")) {

System.out.println("Title is correct.");

} else {

System.out.println("Title is incorrect.");

}

// Close the browser

driver.quit();

}

}

This script opens the Google homepage and checks if the title is "Google."

**3. How do you locate a web element in Selenium?**

**Answer**:  
In Selenium, you can locate web elements using different strategies. For example:

* **By ID**: If the element has an ID attribute.
* **By Name**: If the element has a Name attribute.
* **By XPath**: For more complex element searches.

Example:

WebDriver driver = new ChromeDriver();

WebElement searchBox = driver.findElement(By.id("searchBox"));

This locates an element with the ID "searchBox".

**4. Write a Selenium script to fill a text field and click a button.**

**Answer**: Here’s how you can fill a text field (like a search box) and click a button using Selenium:

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

public class FillForm {

public static void main(String[] args) {

System.setProperty("webdriver.chrome.driver", "path/to/chromedriver");

WebDriver driver = new ChromeDriver();

// Open the website

driver.get("https://www.example.com");

// Locate and fill the text field

WebElement inputField = driver.findElement(By.id("searchInput"));

inputField.sendKeys("Hello Selenium");

// Locate and click the button

WebElement submitButton = driver.findElement(By.id("submitButton"));

submitButton.click();

driver.quit();

}

}

This script fills a search input field and clicks a button after entering text.

**5. What is the difference between findElement() and findElements() in Selenium?**

**Answer**:

* findElement() is used to locate a single element on the page. If the element is not found, it will throw an error.
* findElements() returns a list of elements that match the given criteria. If no elements are found, it returns an empty list.

Example:

WebElement element = driver.findElement(By.id("submitButton")); // For single element

List<WebElement> elements = driver.findElements(By.className("menuItem")); // For multiple elements

**6. How do you handle an alert popup in Selenium?**

**Answer**:  
In Selenium, you can switch to an alert using the Alert interface and perform actions like accepting or dismissing the alert.

Example:

Alert alert = driver.switchTo().alert();

alert.accept(); // To accept the alert

// OR

alert.dismiss(); // To dismiss the alert

**7. How can you perform a mouse hover action in Selenium?**

**Answer**:  
To perform a mouse hover action, you can use the Actions class in Selenium.

Example:

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.interactions.Actions;

public class MouseHover {

public static void main(String[] args) {

System.setProperty("webdriver.chrome.driver", "path/to/chromedriver");

WebDriver driver = new ChromeDriver();

// Open the website

driver.get("https://www.example.com");

// Locate the element to hover over

WebElement menu = driver.findElement(By.id("menu"));

// Perform mouse hover

Actions actions = new Actions(driver);

actions.moveToElement(menu).perform();

driver.quit();

}

}

This will hover over the "menu" element on the page.

**8. How can you handle drop-down menus in Selenium?**

**Answer**:  
To handle drop-downs in Selenium, you can use the Select class to interact with <select> elements.

Example:

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.support.ui.Select;

public class DropdownExample {

public static void main(String[] args) {

System.setProperty("webdriver.chrome.driver", "path/to/chromedriver");

WebDriver driver = new ChromeDriver();

// Open the website

driver.get("https://www.example.com");

// Locate the dropdown

Select dropdown = new Select(driver.findElement(By.id("dropdownMenu")));

// Select an option by visible text

dropdown.selectByVisibleText("Option 1");

driver.quit();

}

}

**9. How do you take a screenshot in Selenium WebDriver?**

**Answer**:  
To take a screenshot, you can use the TakesScreenshot interface.

Example:

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.OutputType;

import org.openqa.selenium.TakesScreenshot;

import java.io.File;

import org.apache.commons.io.FileUtils;

public class ScreenshotExample {

public static void main(String[] args) throws Exception {

System.setProperty("webdriver.chrome.driver", "path/to/chromedriver");

WebDriver driver = new ChromeDriver();

// Open the website

driver.get("https://www.example.com");

// Take a screenshot

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

// Save the screenshot

FileUtils.copyFile(screenshot, new File("screenshot.png"));

driver.quit();

}

}

10. **What is the purpose of WebDriverWait in Selenium?**

**Answer**:  
WebDriverWait is used to wait for a specific condition (such as visibility or clickability) to be true before proceeding with further actions. It helps ensure that the elements you're interacting with are ready to be used, especially in cases where page loading or AJAX calls might delay their availability.

Example:

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.By;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.ui.WebDriverWait;

import org.openqa.selenium.support.ui.ExpectedConditions;

import java.time.Duration;

public class WaitExample {

public static void main(String[] args) {

System.setProperty("webdriver.chrome.driver", "path/to/chromedriver");

WebDriver driver = new ChromeDriver();

// Open the website

driver.get("https://www.example.com");

// Wait for an element to be clickable

WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));

WebElement button = wait.until(ExpectedConditions.elementToBeClickable(By.id("submitButton")));

// Click the button after it becomes clickable

button.click();

driver.quit();

}

}

Sure! Here are **20 more Selenium with Java interview questions and answers** designed to help beginners and laymen understand key concepts of Selenium automation testing:

**11. How do you navigate back and forward in Selenium WebDriver?**

**Answer**:  
You can use the navigate() method to go back and forward in the browser history.

Example:

WebDriver driver = new ChromeDriver();

driver.get("https://www.example.com");

driver.navigate().to("https://www.google.com"); // Navigate to Google

driver.navigate().back(); // Go back to the previous page (example.com)

driver.navigate().forward(); // Go forward to Google again

**12. What is the Actions class used for in Selenium?**

**Answer**:  
The Actions class is used to perform complex user interactions, like mouse movements, drag and drop, and keyboard actions.

Example:

Actions actions = new Actions(driver);

actions.moveToElement(driver.findElement(By.id("menu"))).perform(); // Hover over menu

**13. How do you switch between multiple tabs in Selenium WebDriver?**

**Answer**:  
You can switch between multiple tabs by using window handles. Each tab has a unique identifier.

Example:

String mainWindow = driver.getWindowHandle();

Set<String> allWindows = driver.getWindowHandles();

for (String window : allWindows) {

if (!window.equals(mainWindow)) {

driver.switchTo().window(window);

break;

}

}

**14. How do you handle dynamic elements in Selenium (elements whose properties change frequently)?**

**Answer**:  
Use dynamic locators such as XPath or CSS selectors based on attributes that change less frequently, or use WebDriverWait to wait for an element to be available before interacting with it.

Example using WebDriverWait:

WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));

WebElement dynamicElement = wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath("//div[contains(text(), 'Dynamic Text')]")));

**15. How do you handle dropdowns in Selenium?**

**Answer**:  
Use the Select class in Selenium to interact with dropdown elements.

Example:

Select dropdown = new Select(driver.findElement(By.id("dropdown")));

dropdown.selectByVisibleText("Option 1");

**16. How do you handle file uploads in Selenium?**

**Answer**:  
You can use the sendKeys() method to upload files if the webpage contains an input field for files.

Example:

WebElement uploadElement = driver.findElement(By.id("fileUpload"));

uploadElement.sendKeys("C:/path/to/file.txt");

**17. How can you perform a right-click (context click) in Selenium?**

**Answer**:  
Use the Actions class to simulate a right-click action.

Example:

Actions actions = new Actions(driver);

WebElement element = driver.findElement(By.id("rightClickElement"));

actions.contextClick(element).perform(); // Right-click

**18. How do you capture a screenshot in Selenium WebDriver?**

**Answer**:  
Use the TakesScreenshot interface to capture screenshots.

Example:

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

FileUtils.copyFile(screenshot, new File("screenshot.png"));

**19. What is the difference between driver.close() and driver.quit() in Selenium?**

**Answer**:

* driver.close(): Closes the current browser window.
* driver.quit(): Closes all browser windows opened by WebDriver and ends the WebDriver session.

**20. How do you handle alerts in Selenium WebDriver?**

**Answer**:  
You can handle alerts using the Alert interface. Alerts can be accepted, dismissed, or used to send text.

Example:

Alert alert = driver.switchTo().alert();

alert.accept(); // Accept the alert

// OR

alert.dismiss(); // Dismiss the alert

**21. How do you handle multiple windows in Selenium WebDriver?**

**Answer**:  
Switch between different windows using getWindowHandles() and switchTo().window().

Example:

String mainWindow = driver.getWindowHandle();

Set<String> allWindows = driver.getWindowHandles();

for (String window : allWindows) {

if (!window.equals(mainWindow)) {

driver.switchTo().window(window);

break;

}

}

**22. How can you read data from Excel files in Selenium?**

**Answer**:  
You can use Apache POI library to read data from Excel files.

Example:

FileInputStream fis = new FileInputStream("data.xlsx");

Workbook wb = new XSSFWorkbook(fis);

Sheet sheet = wb.getSheetAt(0);

Row row = sheet.getRow(0);

String cellData = row.getCell(0).getStringCellValue();

**23. How do you use JavaScriptExecutor in Selenium?**

**Answer**:  
JavaScriptExecutor allows you to execute JavaScript code in the browser through Selenium.

Example:

JavascriptExecutor js = (JavascriptExecutor) driver;

js.executeScript("window.scrollBy(0,1000)"); // Scroll down by 1000 pixels

**24. How do you handle a page that contains AJAX calls in Selenium?**

**Answer**:  
Use WebDriverWait to wait for elements to load after AJAX calls.

Example:

WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));

WebElement element = wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("element\_id")));

**25. How do you execute tests in parallel using TestNG?**

**Answer**:  
You can use the parallel attribute in TestNG XML configuration to run tests or test methods in parallel.

Example:

<suite name="Test Suite" parallel="methods" thread-count="2">

<test name="Test1">

<classes>

<class name="TestClass1"/>

</classes>

</test>

</suite>

**26. What is ImplicitWait in Selenium?**

**Answer**:  
ImplicitWait tells WebDriver to wait for a certain amount of time before throwing an exception if an element is not found.

Example:

driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

**27. How do you perform a drag-and-drop action in Selenium?**

**Answer**:  
Use the Actions class to perform a drag-and-drop action.

Example:

WebElement source = driver.findElement(By.id("source"));

WebElement target = driver.findElement(By.id("target"));

Actions actions = new Actions(driver);

actions.dragAndDrop(source, target).perform();

**28. How do you interact with hidden elements in Selenium?**

**Answer**:  
You can use JavaScript to interact with hidden elements that are not directly clickable in Selenium.

Example:

JavascriptExecutor js = (JavascriptExecutor) driver;

js.executeScript("arguments[0].click();", hiddenElement);

**29. What is the purpose of WebDriverWait in Selenium?**

**Answer**:  
WebDriverWait is used to wait for specific conditions (like visibility, presence, or clickability) to be true before performing actions.

Example:

WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));

WebElement element = wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("element\_id")));

**30. What is the Page Object Model (POM) in Selenium?**

**Answer**:  
The Page Object Model (POM) is a design pattern in Selenium that encourages creating a separate class for each web page. This makes the code more organized and easier to maintain.

**31. How do you handle SSL certificate errors in Selenium?**

**Answer**:  
You can configure the browser to ignore SSL certificate errors by using browser-specific options.

Example:

ChromeOptions options = new ChromeOptions();

options.addArguments("--ignore-certificate-errors");

WebDriver driver = new ChromeDriver(options);

**32. How do you switch to an iframe in Selenium?**

**Answer**:  
You can switch to an iframe by using the switchTo().frame() method.

Example:

driver.switchTo().frame("frameName");

**33. How do you take a screenshot in Selenium when an exception occurs?**

**Answer**:  
You can capture a screenshot using the TakesScreenshot interface in a try-catch block.

Example:

try {

// Some test steps

} catch (Exception e) {

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

FileUtils.copyFile(screenshot, new File("error\_screenshot.png"));

}

**34. What is a test framework, and why is it important in Selenium?**

**Answer**:  
A test framework provides structure to your test scripts. It helps in organizing the code, executing tests, and generating reports. Common test frameworks used with Selenium include **TestNG** and **JUnit**.

**35. How do you handle a checkbox in Selenium?**

**Answer**:  
You can interact with checkboxes by using the click() method, and you can verify whether the checkbox is checked or not using isSelected().

Example:

WebElement checkbox = driver.findElement(By.id("checkbox"));

if (!checkbox.isSelected()) {

checkbox.click(); // Check the box

}

**36. What is the purpose of the clear() method in Selenium WebDriver?**

**Answer**:

The clear() method is used to clear the text or value inside a text field or text area before entering new text.

Example:

WebElement textField = driver.findElement(By.id("username"));

textField.clear(); // Clears any existing text in the text field

textField.sendKeys("new\_username");

**37. How do you handle a drop-down with multiple selections in Selenium?**

**Answer**:

If a dropdown allows multiple selections, use the Select class to handle it and select multiple options.

Example:

Select dropdown = new Select(driver.findElement(By.id("multiSelect")));

dropdown.selectByVisibleText("Option 1");

dropdown.selectByVisibleText("Option 2");

**38. How can you execute a JavaScript alert and handle its response in Selenium?**

**Answer**:

Selenium can handle JavaScript alerts using the Alert interface. You can either accept or dismiss the alert.

Example:

Alert alert = driver.switchTo().alert();

alert.accept(); // To click "OK" on the alert

**39. How can you wait for an element to be clickable in Selenium?**

**Answer**:

You can use WebDriverWait with ExpectedConditions to wait for an element to become clickable.

Example:

WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));

WebElement button = wait.until(ExpectedConditions.elementToBeClickable(By.id("submitButton")));

button.click(); // Perform click after waiting for it to be clickable

**40. How do you get the current URL of a webpage in Selenium?**

**Answer**:

You can use the getCurrentUrl() method to get the URL of the current webpage.

Example:

String currentUrl = driver.getCurrentUrl();

System.out.println("Current URL: " + currentUrl);

**41. How do you handle unexpected pop-up windows in Selenium?**

**Answer**:

You can switch to the pop-up window using driver.switchTo().window() and close it or interact with it.

Example:

String parentWindow = driver.getWindowHandle();

Set<String> allWindows = driver.getWindowHandles();

for (String window : allWindows) {

if (!window.equals(parentWindow)) {

driver.switchTo().window(window); // Switch to the new window

driver.close(); // Close the pop-up window

break;

}

}

driver.switchTo().window(parentWindow); // Switch back to the parent window

**42. How do you perform mouse movement actions like drag and drop in Selenium?**

**Answer**:

You can perform drag-and-drop actions using the Actions class in Selenium.

Example:

Actions actions = new Actions(driver);

WebElement source = driver.findElement(By.id("source"));

WebElement target = driver.findElement(By.id("target"));

actions.dragAndDrop(source, target).perform();

**43. How do you handle an element that is not immediately visible or interactable in Selenium?**

**Answer**:

You can use WebDriverWait and ExpectedConditions to wait for the element to become visible or interactable before performing any action.

Example:

WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));

WebElement element = wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("elementId")));

element.click(); // Perform the action after the element is visible

**44. What are findElement() and findElements() in Selenium, and how do they differ?**

**Answer**:

* findElement(): Returns the first element matching the locator. If no element is found, it throws an exception.
* findElements(): Returns a list of all elements matching the locator. If no elements are found, it returns an empty list.

Example:

WebElement element = driver.findElement(By.id("submitButton"));

List<WebElement> elements = driver.findElements(By.className("menuItem"));

**45. How do you handle dynamic elements whose IDs change every time in Selenium?**

**Answer**:

You can use relative XPath or CSS selectors with attributes that don't change, such as name, class, or text().

Example using XPath:

WebElement element = driver.findElement(By.xpath("//button[text()='Submit']"));

**46. What is the difference between Thread.sleep() and WebDriverWait in Selenium?**

**Answer**:

* Thread.sleep() causes the thread to pause for a specific amount of time, which is often not ideal because it introduces unnecessary delays.
* WebDriverWait waits for a specific condition (e.g., visibility of an element) to be met, and it only waits as long as necessary.

Example using WebDriverWait:

WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));

WebElement element = wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("elementId")));

**47. How do you handle a checkbox that needs to be checked if it is not already checked in Selenium?**

**Answer**:

You can check if the checkbox is already selected using isSelected(). If it’s not selected, you can click it.

Example:

WebElement checkbox = driver.findElement(By.id("checkbox"));

if (!checkbox.isSelected()) {

checkbox.click(); // Click to check the box

}

**48. How do you run Selenium tests in parallel using TestNG?**

**Answer**:

To run tests in parallel, you need to configure the parallel attribute in the TestNG XML file.

Example:

<suite name="Selenium Suite" parallel="tests" thread-count="2">

<test name="Test1">

<classes>

<class name="TestClass1"/>

</classes>

</test>

<test name="Test2">

<classes>

<class name="TestClass2"/>

</classes>

</test>

</suite>

**49. How do you select an option from a radio button group in Selenium?**

**Answer**:

You can select a radio button by clicking the WebElement that corresponds to the option you want to select.

Example:

WebElement radioButton = driver.findElement(By.id("radioOption"));

if (!radioButton.isSelected()) {

radioButton.click(); // Select the radio button

}

**50. How do you retrieve the text from a WebElement in Selenium?**

**Answer**:

You can use the getText() method to retrieve the text of a WebElement.

Example:

WebElement element = driver.findElement(By.id("textElement"));

String text = element.getText();

System.out.println("Text is: " + text);

**51. How do you switch to a different frame in Selenium?**

**Answer**:

To switch to a frame, you can use the switchTo().frame() method with the frame's index, name, or WebElement.

Example:

driver.switchTo().frame("frameName"); // Switch to a frame by name

// OR

driver.switchTo().frame(0); // Switch to the first frame by index

**52. How can you scroll down a page in Selenium WebDriver?**

**Answer**:

You can use JavaScriptExecutor to scroll down a webpage.

Example:

JavascriptExecutor js = (JavascriptExecutor) driver;

js.executeScript("window.scrollBy(0, 1000);"); // Scroll down by 1000 pixels

**53. How do you handle an element that is inside a shadow DOM in Selenium?**

**Answer**:

Selenium WebDriver does not natively support interacting with shadow DOM elements. However, you can use JavaScriptExecutor to interact with elements inside the shadow DOM.

Example:

JavascriptExecutor js = (JavascriptExecutor) driver;

WebElement shadowRoot = (WebElement) js.executeScript("return document.querySelector('host-element').shadowRoot");

WebElement shadowElement = shadowRoot.findElement(By.id("shadowElementId"));

shadowElement.click();

**Comments**

Popular Posts

[](https://explore2test.blogspot.com/2025/01/1-software-testing-software-testing-is.html)

[ST - Session 1](https://explore2test.blogspot.com/2025/01/1-software-testing-software-testing-is.html)

*Session - 1 Software Testing Software testing is a crucial process in the software development life cycle (SDLC). It involves evaluating a software application to ensure it meets the required standards. Roles and Responsibilities Testers/QA/SDET/PQE: These roles are responsible for testing software applications. SDET (Software Development Engineer in Test): An SDET is a skilled tester who can write code and develop automated tests. PQE (Principle Quality Engineer): A PQE is a senior-level quality engineer who leads testing teams and ensures the overall quality of software applications. Software Development Engineering Test SDET/PQE Levels: SDET-1: Junior-level SDET SDET-11: Mid-level SDET SDET-111: Senior-level SDET PQE PQE-1: Junior-level PQE PQE-11: Mid-level PQE PQE-111: Senior-level PQE Software Application A software application is a collection of programs designed to achieve specific functionalities. It consists of various components, including: User Interface (UI): The UI is the...*

[ST - Session 2](https://explore2test.blogspot.com/2025/01/st-session-2.html)

*Session 2 : Software Testing Software testing is a crucial part of the Software Development Life Cycle (SDLC). It ensures that the software works correctly and meets the required standards. Why Test Software? There are two main types of software testing: Functional Testing: This type of testing checks if the software works as expected. It involves testing the User Interface (UI), Application Programming Interface (API), and Database (DB). Non-Functional Testing: This type of testing checks the software's performance, compatibility, security, and accessibility. How to Test Software? There are two ways to test software: Manual Testing: This involves testing the software manually, without using any automated tools. Automation Testing: This involves using automated tools to test the software. Automation testing can be done using various programming languages and tools like Selenium, Java, and Python. Software Testing Life Cycle (STLC) The STLC is a process that involves several stages:...*

[ST - Session 5](https://explore2test.blogspot.com/2025/01/st-session-5.html)

*Session 5 SDLC (Software Development Life Cycle) : This is the process that software goes through from start to finish. It includes different stages like gathering requirements, designing the software, coding it, testing it, and releasing it. 3 Types of SDLC Approaches Old / Legacy Methodologies Testing happens only after development . The stages of the SDLC are: RG (Requirement Gathering) : Collecting what the client wants. A (Analysis) : Understanding how to make the requirements work. D (Design) : Creating a blueprint of how the app will look and function. C (Coding/Development) : Writing the actual code. T (Testing) : Checking if the software works as expected. R (Release) : Deploying the software for use. M (Maintenance) : Ongoing support after the software is released. New Methodologies Testing happens at every stage . For example: RG : Testing is done right after gathering requirements. A : Testing checks if the analysis makes sense. D : Testing is don...*

[Powered by Blogger](https://www.blogger.com/)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Interview Questions from my last 3 Interviews Experience July 2024  
  
1) Java program to remove duplicates characters from given String.   
2) Program Remove the second highest element from the HashMap.   
3) Java program to Generate prime numbers between 1 & given 4  
number   
4) How to find the missing values from a sorted array.   
5) Java program to input name, middle name and surname of a   
person and print only the initials.   
5) Program to Print all Treemap elements?   
6) What is a singleton Design Pattern? How do you implement that in your framework?   
7) Write the Top 5 test cases for Booking Coupons.   
8) What is serialization and deserialization?   
9) What is the Difference between status codes 401 and 402? 1  
10) Difference between selenium 3 and selenium 4?   
11) What is delegate in Java and where do you use Delegate in your Framework?   
12) How many maximum thread-pool can you open in the TestNG?   
13) What are the Major challenges that come into the picture when you do parallel testing using TestNG and Grid?   
14) How do you integrate your automation framework with the Jenkins pipeline?   
15) What will happen if we remove the main method from the java program?   
16) What is the component of your current Project?   
17) How do you pass parameters in TestNG?   
18) Write the logic of retrying the failed test case with a minimum 3 numbers of time in Automation Testing. Which Interface do you use for it?   
19) What is the OOPs concept in java?   
20) What is encapsulation?   
21) What is Polymorphism?   
22) Difference Between Classes and Objects?   
23) What is collection in Java?   
24) What is out in System.out.println?   
25) In How many ways can we create an object?   
26) Why is Java not 100% Object-oriented?  
27) Can we make a constructor as Static?   
28) How to convert a JSON object to a java object using Jackson?   
29) What is the difference between Abstraction Class and Interfaces?   
30) Difference between String, StringBuilder, and Stringbuffer?   
31) What are other immutable classes in Java apart from String?   
32) Difference between TreeMap and HashMap?   
33) How do you set priorities for test automation, which test needs to be automated first?   
34) How do you set test case priorities for your team?   
35) What are the functional things you need to test on e-commerce sites?