What are the Access Modifiers in Java?

1. Access modifiers in Java help to restrict the scope of a class, constructor, variable, method, or data member.
2. **Types of Access Modifiers:** Default, Private, Protected and Public.

What are the various types of Classes in Java?

1. Static, Final, Abstract, Concrete and Singleton Class.

What is Singleton Class in Java?

1. In object-oriented programming, a singleton class is a class that can have only one object (an instance of the class) at a time.

Java OOPs Concepts:

1. **Inheritance:** it is a mechanism in java by which one class is allowed to inherit the features (fields and methods) of another class.
2. **Encapsulation:** it is a process of wrapping code and data together into a single unit, for example, a capsule which is mixed of several medicines. **The public setXXX() and getXXX() methods are the access points of the instance variables of the EncapTest class.**
3. **Abstraction:** it is the process of hiding certain details and showing only essential information to the users. **In java, abstraction is achieved by interfaces and abstract classes.**
4. **Polymorphism:** it means having many forms. Using polymorphism, we can perform a single action in different ways. **It could be achieved by method overloading and overriding (Runtime Polymorphism).**

What is a static keyword in Java?

1. In Java, if we want to access class members, we must first create an instance of the class. But there will be situations where we want to access class members without creating any object.
2. In those situations, we can use the static keyword in Java.
3. **The static can be:**
    a. **Variable (also known as a class variable):** if we declare a variable static, all objects of the class share the same static variable.
    b. **Method (also known as a class method):** we can invoke static methods directly using the class name.
    c. **Block:** static blocks are used to initialize the static variables.
        i. // static variable: static int age;
        ii. // static block: static {age = 23; }
    d. **Nested class**

What is an Array in Java?

1. Java array is an object which contains elements of a similar data type.
2. int Array = new int[20];
3. int[][] intArray = new int[10][20];

What is Java Arraylist?

1. The ArrayList class is a resizable array, which can be found in the java.util package.
2. ArrayList<String> cars = new ArrayList<String>();
3. Sort an arraylist: collections.sort(arraylist name);

What is a Java Constructor?

1. A constructor in Java is a special method that is used to initialize objects.
2. The constructor's name must match the class name and cannot have a return type (like void).
3. The constructor is called when the object is created.
4. **Types of Constructors:** Default and Parameterized.

What is Java Super Keyword?

1. The super keyword refers to superclass (parent) objects.
2. It is used to call superclass methods, and to access the superclass constructor.

What is Java this Keyword?

1. This keyword refers to the current object in a method or constructor.
2. The most common use of this keyword is to eliminate the confusion between class attributes and parameters with the same name (because a class attribute is shadowed by a method or constructor parameter).

What is the Final Keyword in Java?

1. The final keyword in java is used to restrict the user. The java final keyword can be used in many contexts. The Final can be:
    a. **Variable:** If you make any variable as final, you cannot change the value of final variable (It will be constant).
    b. **Method:** If you make any method as final, you cannot override it.
    c. **Class:** If you make any class as final, you cannot extend it.

What is Exception Handling in Java?

1. The Exception Handling in Java is one of the powerful mechanism to handle the runtime errors so that the normal flow of the application can be maintained.
2. **Java Exception Keywords:**
    a. **Try:** The "try" keyword is used to specify a block where we should place an exception code.
    b. **Catch:** The "catch" block is used to handle the exception.
    c. **finally:** The "finally" block is used to execute the necessary code of the program. It is executed whether an exception is handled or not.

What is the difference between Throw and Throws keywords in Java?

1. **Throw:** Java throw keyword is used throw an exception explicitly in the code, inside the function or the block of code. Throw is used within the method.

2. **Throws**: Java throws keyword is used in the method signature to declare an exception which might be thrown by the function while the execution of the code. Throws is used with the method signature.

What is finalize() method in Java?

1. The finalize() method is used just before object is destroyed and can be called just prior to object creation.

What are Java Wrapper Classes?

1. Wrapper classes provide a way to use primitive data types (int, boolean, etc..) as objects.
2. ArrayList<Integer> myNumbers = new ArrayList<Integer>();

Can We Override Static Method in Java? If not, Why?

1. No, we cannot override static methods because methods overriding is based on dynamic binding at runtime and the static methods are bonded using static binding at compile time. So, we cannot override static methods.

What are collections in Java?

1. The Collection in Java is a framework that provides an architecture to store and manipulate the group of objects.
2. Java Collection means a single unit of objects. Java Collection framework provides many interfaces (Set, List, Queue, Deque) and classes (ArrayList, Vector, LinkedList, PriorityQueue, HashSet, LinkedHashSet, TreeSet).

What is the difference between Collection and Collections?

1. **Collection:** It is an interface.
2. **Collections:** It is a utility class.

What is the difference between Set and List in Java?

1. The list implementation allows us to add the same or duplicate elements.
2. The set implementation does not allow us to add the same or duplicate elements.

What is Hashing in Java?

1. Hashing is the process of mapping the data to some representative integer value using the concept of hashing algorithms.
2. In Java, a hash code is an integer value that is linked with each object.
3. Hashing finds its data structure implementation in HashTables and HashMaps.

What is Hash Map?

1. It stores elements in key/value pairs.
2. Here, keys are unique identifiers used to associate each value on a map.
3. It allows one null for key and multiple nulls for values.
4. It uses **put method** to insert a new element.

What is Hash Table?

5. It stores elements in key/value pairs, and it uses **put method** to insert a new element.
1. It does not allow null for key as well as for value.

What is Hash Set?

1. It does not allow duplicates.
2. It can have a single null value. It uses **add method.**

What is the difference between Integer.parseInt() and Integer.valueOf()?

1. **parseInt():** will be returning the primitive type int.
2. **valueOf():** will be returning the Integer wrapper Object.

Java Program to "Sort an Array":

```
1.  public class Main {
2.     public static void main(String[] args) {
3.        //define original array
4.        int [] intArray = new int []
    {52,45,32,64,12,87,78,98,23,7};
5.        int temp = 0;
6.        //print original array
7.        System.out.println("Original array: ");
8.        for (int i = 0; i <intArray.length; i++) {
9.           System.out.print(intArray[i] + " ");
10.       }
11.       //Sort the array in ascending order using two for loops
12.       for (int i = 0; i <intArray.length; i++) {
13.          for (int j = i+1; j <intArray.length; j++) {
14.             if(intArray[i] >intArray[j]) {      //swap elements if
    not in order
15.                temp = intArray[i];
16.                intArray[i] = intArray[j];
17.                intArray[j] = temp;
18.             }
19.          }
20.       }
21.       //print sorted array
22.       System.out.println("\nArray sorted in ascending
    order: ");
23.       for (int i = 0; i <intArray.length; i++) {
24.          System.out.print(intArray[i] + " ");
25.       }
26.    }
```

Java Program to "Split a String":

```
1.  public class SplitExample{
2.  public static void main(String args[]){
3.  String s1="java string split method by javatpoint";
4.  String[] words=s1.split("\\s");//splits the string based on
    whitespace
5.  //using java foreach loop to print elements of string array
```

```
6.  for(String w:words){
7.  System.out.println(w);
8.  }}}
```

Java Program to "Reverse a String":

```
1.  import java.io.*;
2.  import java.util.Scanner;
3.  class GFG {
4.     public static void main (String[] args) {
5.        String str= "Geeks", nstr="";
6.        char ch;
7.      System.out.print("Original word: ");
8.      System.out.println("Geeks"); //Example word
9.      for (int i=0; i<str.length(); i++)
10.     {
11.       ch= str.charAt(i); //extracts each character
12.       nstr= ch+nstr; //adds each character in front of the
    existing string
13.     }
14.     System.out.println("Reversed word: "+ nstr);
15.   }}
```

Java Program to "Reverse any String":

```
1.  class ReverseString {
2.     public static void main(String[] args)
3.     {
4.        String input = "Geeks for Geeks";
5.        StringBuilder input1 = new StringBuilder();
6.        input1.append(input);
7.        input1.reverse();
8.        System.out.println(input1); }}
```

What is StringBuilder?

1. StringBuilder in Java is a class used to create a mutable, or in other words, a modifiable succession of characters.

What is StringBuffer?

1. The StringBuffer class is used to create mutable string. It is same as String class except it is mutable and thread safe.

What is the difference between String, StringBuilder and StringBuffer?

1. If a string can change and will be accessed from multiple threads, use a StringBuffer because StringBuffer is synchronous, so you have thread-safety.
2. If you don't want thread-safety than you can also go with StringBuilder class as it is not synchronized.

Java Program to "Find the Largest and Second Largest Number in a given Array":

```
1.  public class findElement
2.  {
3.     public static void main(String []args)
4.     {
5.  int [] arrayname = new int [] {1, 2, 3, 4, 2, 7, 8, 8, 3};
6.  int temp, size;
7.  size = arrayname.length;
8.        for(int i=0;i<size;i++)   //Use to hold an element
9.        {
10.          for(int j=i+1;j<size;j++)   //Use to check for rest of
    the elements
11.          {
12.             if(arr[i]<arr[j])    //Compare and swap
13.             {
14.                int temp=arr[i];
15.                arr[i]=arr[j];
16.                arr[j]=temp;
17. }}}
18.       System.out.println("Largest element is
    "+arrayname[size-1]);  //Display Largest
19.    }}
```

Java program to "Extract a word from a string":

```
1.  import java.util.Scanner;
2.  public class string {
3.     public static void main(String args[]){
4.        Scanner sc  = new Scanner(System.in);
5.        String sentence = "This is a bird";
6.        System.out.println("Enter a word to extract from the
    string: ");
7.        String wordToextract = sc.next();
8.        if(sentence.contains(wordToextract)){
9.           int y = sentence.indexOf(wordToextract);
10.          System.out.println(y);
11.          String u = sentence.substring(y ,
    y+(wordToextract.length()));
12.          System.out.println(u);
13.       } }}
```

Java Program to illustrate a Set and Iterator:

```
1.  import java.util.*;
2.  public class SetDemo {
3.     public static void main(String args[])
4.     {
5.        Set<String> set = new HashSet<String>();
6.        set.add("Welcome");
7.        set.add("To");
8.        set.add("Geeks");
9.        set.add("4");
10.       set.add("Geeks");
11.       System.out.println("Set: " + set);
12.       Iterator value = set.iterator();
13.       System.out.println("The iterator values are: ");
14.       while (value.hasNext()) {
15.          System.out.println(value.next());
16.       }}}
```

Java program to "Find Duplicate elements in an Array":

```
1.  public class DuplicateElement {
2.     public static void main(String[] args) {
3.        //Initialize array
4.        int [] arr = new int [] {1, 2, 3, 4, 2, 7, 8, 8, 3};
5.        System.out.println("Duplicate elements in given
    array: ");
```

```
        //Searches for duplicate element
        for(int i = 0; i < arr.length; i++) {
            for(int j = i + 1; j < arr.length; j++) {
                if(arr[i] == arr[j])
                    System.out.println(arr[j]);
            }
        }}}
```

Java program to “find the duplicate characters in a string”:

```
public class DuplicateCharacters {
    public static void main(String[] args) {
        String string1 = "Great responsibility";
        int count;
        //Converts given string into character array
        char string[] = string1.toCharArray();
        System.out.println("Duplicate characters in a given
string: ");
        //Counts each character present in the string
        for(int i = 0; i <string.length; i++) {
            count = 1;
            for(int j = i+1; j <string.length; j++) {
                if(string[i] == string[j] && string[i] != ' ') {
                    count++;
                    //Set string[j] to 0 to avoid printing visited
character
                    string[j] = '0';
                }}
            //A character is considered as duplicate if count is
greater than 1
            if(count > 1 && string[i] != '0')
                System.out.println(string[i]);
              }}}
```

What are the various stages of the Automation Testing Life Cycle?

1. Deciding the scope of Test Automation
2. Choosing the right Automation Tool
3. Plan, Design, and Strategy
4. Set-Up Test Environment
5. Test Script & Execution
6. Examine and Maintenance Approach

When Should a Test Case Be Automated?

1. The task is going to be repeated.
2. It's going to save time.
3. The requirements, the test, or the task are low risk, stable, and unlikely to change often.
4. The test is subject to human error.
5. The test is time consuming.
6. The test has significant downtime between steps.
7. The test is repetitive.

How to launch chrome browser in Selenium?

1. WebDriver driver = new ChromeDriver();
2. System.setProperty("webdriver.chrome.driver","D:\\ChromeDriver\\chromedriver.exe");
3. WebDriver is an interface.
4. **Why we wont prefer ChromeDriver driver = new ChromeDriver()?**
    a. The ChromeDriver instance which gets created based on above statement will be only able to invoke and act on the methods implemented by ChromeDriver and supported by Chrome Browser only.

What are the various types of Locators in Selenium?

1. ID, Class Name, Name, Xpath, CSS Selector, Link Text, Partial Link Text, and HTML Tag Name.

XPath Axes for Dynamic XPath In Selenium:

1. ancestor
2. ancestor-or-self
3. descendant
4. descendant-or-self
5. preceding sibling
6. following-sibling
7. parent
8. child
9. following
10. preceding
11. **Examples:**
    a. //label[text()='Email']/following-sibling::input[1]
    b. //td[text()='Maria Anders']/preceding-sibling::td/child::input
    c. //label[text()='Email']/following-sibling::input[1]/parent::div
    d. //div[@class='container']/child::input[@type='text']
    e. //div[@class='container']/descendant::button
    f. //div[@class='buttons']/ancestor-or-self::div
    g. //label[text()='Password']/following::input[1]

How to find Xpath using "contains" keyword?

1. //input[contains(@id, "test_"]
2. //div[@id=""]//a[contains(text(), "")]

How to handle static dropdown in Selenium?

1. Select s = new Select(driver.findElement(By.id("<< id exp>>")));
2. s.selectByVisibleText("Selenium");
3. s.selectByIndex(1);
4. s.selectByValue("Testing");

How to handle dyanamic dropdown in Selenium?

1. List <WebElements> Elements = driver.findelements (By.xapth ="")
2. for (WebElements e:elements) {
    a. if (e.getText(). equalsIgnoreCase ("India") {
    b. e.click();
    c. break }}

What is JavaScriptExecutor in Selenium?

1. JavaScriptExecutor is an interface provided by Selenium Webdriver, which presents a way to execute JavaScript from Webdriver.

2. **Click using JavaScriptExecutor**
    a. JavascriptExecutor js = (JavascriptExecutor) driver;
    b. js.executeScript("document.getElementById('enter element id').click();");
3. **Enter Data using JavaScriptExecutor**
    a. js.executeScript("document.getElementById(id').value='someValue';");
4. **Scroll using JavaScriptExecutor**
    a. js.executeScript("window.scrollBy(0,500)");

How to perform double click in Selenium?

1. Actions act = new Actions(driver);
2. WebElement ele = driver.findElement(By.xpath("XPath of the element"));
3. act.doubleClick(ele).perform();

How to click on an element which is not clickable in Selenium?

1. WebElement element = driver.findElement(By.id("Login"));
2. Actions act = new Actions(driver);
3. Act.moveToElement(element).click().build().perform();

How to perform SHIFT+CTRL+S in Selenium?

1. Actions act = new Actions(driver);
2. act.keyDown(Keys.SHIFT).keyDown(Keys.CONTROL).sendKeys("s").build().perform();

How to Drag and Drop in Selenium?

1. Actions act = new Actions(driver);

2. Action dragAndDrop = act.clickAndHold(fromElement).moveToElement(toElement).release(toElement).build();
3. dragAndDrop.perform();

How to handle Sliders in Selenium?

1. WebElement Slider = driver.findElement(By.xpath("//*[@id='slider-range']/a[1]"));
2. Thread.sleep(3000);
3. Actions moveSlider = new Actions(driver);
4. Action action = moveSlider.dragAndDropBy(Slider, 30, 0).build(); // 30 and 0 are coordinates.
5. action.perform();

How to handle Checkbox and Radio Button in Selenium?

1. driver. findElement(By.cssSelector("label[for='hobbies-checkbox-1']")).click();
2. **Validations on Checkboxes:**
    a. **isSelected():** Checks whether a checkbox is selected or not.
    b. **isDisplayed():** Checks whether a checkbox displays on the web page or not.
    c. **isEnabled():** Checks whether a checkbox is enabled or not.
3. The same methods apply to radio buttons as well and we can verify them using same methods.

How to print the text of all the links present on the webpage using Selenium?

1. List<WebElement> links = driver.findElements(By.tagname(“a”));
2. System.out.println(links.size());
3. For(int=0;i<links.size();i++){
4. String linktext = links.get(i).getText();
5. System.out.println(linktext);}}

Types of Waits:

1. **Implicit Wait (Entire Duration)**
    a. driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
2. **Explicit Wait (Specific Element)**
    a. WebDriverWait wait = new WebDriverWait(driver,30);
    b. wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath(“”)));
3. **Fluent Wait (Wait and Frequency)**
    a. FluentWait wait = new FluentWait(driver);
    b. wait.withTimeout(5000, TimeUnit.MILLISECONDS);
    c. wait.pollingEvery(250, TimeUnit.MILLISECONDS);
    d. wait.ignoring(NoSuchElementException.class)

How to take Screenshots in Selenium?

1. TakesScreenshot scrShot =((TakesScreenshot)webdriver);
2. File SrcFile=scrShot.getScreenshotAs(OutputType.FILE);
3. File DestFile=new File(fileWithPath);
4. FileUtils.copyFile(SrcFile, DestFile);

How to handle multiple windows/multiple popups in Selenium?

1. String parent=driver.getWindowHandle();
2. Set<String>s=driver.getWindowHandles();
3. Iterator<String> I1= s.iterator();
4. while(I1.hasNext()) {
5. String child_window=I1.next();
6. if(!parent.equals(child_window)) {
7. driver.switchTo().window(child_window);
8. driver.close(); }}
9. driver.switchTo().window(parent);

How to Handle iFrames in Selenium?

1. <iframe src = “URL”></iframe>
2. Driver.SwitchTo().frame(String Name);
3. Driver.SwitchTo().frame(String ID);

How to use Log4j in Selenium?

It provides an Information about what goes over during test execution on every step.

1. public class Myclass {
2. static Logger log = Logger.getLogger(Myclass.class);
3. public static void main(String[] args) {
4. PropertyConfigurator.configure("path\\to\\log4j.properties");
5. driver.get("https://www.browserstack.com/users/sign_in");
6. log.info("Open browserstack");
7. driver.manage().window().maximize();
8. log.info("Maximize window size");
9. driver.close();
10. }}

The above program will run an automation script, and simultaneously create a log file at the specified location with the name given in the log4j.properties file.

What are Listeners in Selenium?

1. Listeners are basically the ones who have the ability to listen to a particular event.
2. It is defined as an interface that modifies the behavior of the system.
3. Listeners allow customization of reports and logs.
4. **Methods of ITestListners:** OnTestStart(), OnTestSuccess(), OnTestFailure(), OnTestSkipped(), OnStart(), OnFinish(), etc.

How to refresh a page in Selenium?

1. driver.navigate().refresh();
2. driver.findElement(By textboxLocator).sendKeys(Keys.F5);
3. driver.findElement(By textboxLocator).sendKeys("\uE035");

How to handle Alerts and Popups in Selenium?

1. Alert alert = driver.switchTo().alert(); // switch to alert
2. driver.switchTo().alert().dismiss();
3. driver.switchTo().alert().accept();
4. driver.switchTo().alert().getText();
5. driver.switchTo().alert().sendKeys("Text");

How to verify color of a web element in Selenium Webdriver?

1. We can verify the color of a webelement in Selenium webdriver using the **getCssValue method** and then pass color as a parameter to it. This returnsthe color in **rgba() format.**
2. Next, we have to use the class Color to convert the rgba() format to Hex. Let us obtain the color an element highlighted in the below image.
3. **Code:**
    a. // identify text
    b. WebElement t = driver.findElement(By.tagName("h1"));
    c. //obtain color in rgba
    d. String s = t.getCssValue("color");
    e. // convert rgba to hex
    f. String c = Color.fromString(s).asHex();
    g. System.out.println("Color is :" + s);

What are the various types of Exceptions in Selenium?

1. #1) org.openqa.selenium.NoSuchElementException
2. #2) org.openqa.selenium.NoSuchWindowException
3. #3) org.openqa.selenium.NoSuchFrameException
4. #4) org.openqa.selenium.NoAlertPresentException
5. #5) org.openqa.selenium.InvalidSelectorException
6. #6) org.openqa.selenium.ElementNotVisibleException
7. #7) org.openqa.selenium.ElementNotSelectableException
8. #8) org.openqa.selenium.TimeoutException
9. #9) org.openqa.selenium.NoSuchSessionException
10. #10) org.openqa.selenium.StaleElementReferenceException

How to Read Excel Files Using Apache POI in Selenium?

1. Download Apache POI jar file.
2. Add download jar files.
3. Create an xlsx file and save it at particular location.
4. FileInputStream fis = new FileInputStream("D:\\Test.xlsx");
5. XSSFWorkbook workbook = new XSSFWorkbook(fis);
6. XSSFSheet sheet = workbook.getSheetAt(0);
7. Row row = sheet.getRow(0);
8. Cell cell = row.getCell(0);
9. System.out.println(sheet.getRow(0).getCell(0));

What is TestNG in Selenium?

1. TestNG is an open-source testing framework where NG stands for 'Next Generation.'
2. It is inspired from JUnit and NUnit but introducing some new functionalities that make it more powerful and easier to use.

What are the advantages of TestNG?

1. It gives the ability to produce **HTML Reports** of execution.
2. **Annotations** made testers life easy.
3. Test cases can be **Grouped & Prioritized** more easily.
4. **Parallel testing** is possible.
5. Generates **Logs.**
6. Data **Parameterization** is possible.
7. **Exclusion and Inclusion** of Test Cases.

How to run the TestNG script?

1. Right-click on the class in Eclipse, click on "Run as" and select "TestNG test".

What are the annotations used in TestNG?

1. **Precondition annotations:** These are the TestNG annotations that are executed before the test.
    a. @BeforeSuite, @BeforeClass, @BeforeTest, @BeforeMethod are the precondition annotations.
2. **Test annotation:** This is the annotation which is only mentioned before the test case (Before the method written to execute the test case).
    a. @Test is the test annotation.
3. **Postcondition annotation:** These are the annotations that are executed after the test case. (After the method is written to execute the test case).
    a. @AfterSuite, @AfterClass, @AfterTest, @AfterMethod are the postcondition annotations.

What is the sequence of execution of the annotations in TestNG?

1. @BeforeSuite
2. @BeforeTest
3. @BeforeClass
4. @BeforeMethod
5. @Test
6. @AfterMethod
7. @AfterClass
8. @Aftertest
9. @AfterSuite

How to set priorities in TestNG?

1. @Test (priority=2)

How will you define grouping in TestNG?

1. @Test(groups="title")

What is a dependency on TestNG?

1. If we want to test any application, and if the login page of the application is not working then we won't be able to test the rest of the scenarios.
2. @Test(dependsOnMethods="LoginTest")

What is timeOut in TestNG?

1. If any method in the script takes a long time to execute, then we can terminate that method using "timeout" in TestNG.
2. @Test(timeout = 5000)

How to disable a test in TestNG?

1. @Test(enabled=”false”)

TestNG Parallel Execution:

```
<suite name = "Parallel Testing Suite">
  <test name = "Parallel Tests" parallel = "methods">
    <classes>
      <class name = "ParallelTest" />
    </classes>
  </test>
</suite>
```

What is Soft and Hart assert in TestNG?

1. **Soft Assert:** if the test case fails, the execution is not terminated when we use soft assert.
    a. softAssert sassert = new softAssert();
    b. sassert.assertAll();
2. **Hard Assert:** If the hard assert fails, then none of the code gets executed after the assert statement.
    a. Assert.assertEquals(actual value, expected value);

What are the common TestNG assertions?

1. Assert.assetEquals(String actual, String expected);
2. Assert.assertTrue(<condition(t/f)>)
3. Assert.assertFalse(<condition(t/f)>)

How to pass parameter in the test case through the testng.xml file? OR how do you set parameters in your Project?

1. **We will have to use the “@parameters” annotation as follows:**
    a. @Parameters({"user_name","password"})
    b. @Test
    c. public void loginapp()
    d. {
    e. driverget(“appname”);
    f. driver.findElement(By.id(“login”)).sendkeys(user_name);
    g. driver.findElement(By.id(“password”)).sendkeys(password);
    h. }
2. **Now, go to the testng.xml file and enter the parameters there as follows:**
    1. <parameter name =”user_name” value=”user1”/>
    2. <parameter password =”password” value =”pass1”/>

What is the need to create a testng.xml file?

1. When we test a project using Selenium Webdriver, it has a lot of classes on it. We cannot choose these classes one by one and put them for automation. Hence we need to create a suite so that all the classes run in a single test suite.
    a. <suite name="Testing Google Apps">
    b. <test name="Regression">
    c. <classes>
    d. <class name="Googletest.GmailTest"/>
    e. <class name="Googletest.MapsTest"/>
    f. <class name="Googletest.ImagesTest"/>
    g. </classes>
    h. </test> <!-- Test -->
    i. </suite> <!-- Suite -->

How to execute failed test cases in TestNG?

1. Right click on the project and choose Refresh. On refreshing, the tester will see a test-output.
2. In the testng.xml file, the suite name is Suite. Therefore, in the above snapshot, there is a folder called Suite. Expand that to get a file called testng-failed.xml.
3. Re-run testng-failed.xml.

How to achieve cross browser testing using selenium?

1. **Step1:** Test cases can be automated using Internet Explorer, Firefox, Chrome, Safari browsers with the help of Selenium WebDriver.
2. **Step 2:** To execute test cases with different browsers in the same machine at the same time a TestNG framework can be integrated with Selenium WebDriver.
3. **Step3:** Write the test cases. The article features code that will test the Browserstack home page on three different browsers – Chrome, Edge, and Firefox.

Defining Browser

```
@BeforeTest
@Parameters("browser")
public void setup(String browser) throws Exception{
//Check if parameter passed from TestNG is 'firefox'
if(browser.equalsIgnoreCase("firefox")){
//create firefox instance
System.setProperty("Path of your gecko driver");
driver = new FirefoxDriver();
}
//Check if parameter passed as 'chrome'
else if(browser.equalsIgnoreCase("chrome")){
//set path to chromedriver.exe
System.setProperty("Path of your chrome driver");
driver = new ChromeDriver();
}
else if(browser.equalsIgnoreCase("Edge")){
//set path to Edge.exe
System.setProperty("Path of edge driver”);
driver = new EdgeDriver();
}
```

Defining TestNG file

```
<suite name="TestSuite" thread-count="2" parallel="tests" >
<test name="ChromeTest">
<parameter name="browser" value="Chrome"/>
<classes>
<class name="test.CrossBrowserTestingScript">
</class>
</classes>
</test>
</suite>
```

What is the difference between Page Object Model and Page Factory?

1. **Page Object Model**
    a. POM is a design pattern.
    b. In POM, one needs to initialize every page object individually.

    c. POM = Page Class (Locators) + Test Class (Test Scripts)
2. **Page Factory**
    a. Page Factory is a class that provides the implantation of the POM design patter.
    b. In Page Factory, all page objects are initialize by using the **initElements()** method.

Why POM?

1. Easy to Maintain
2. Easy Readability of Scripts
3. Reduce duplicacy
4. Re-usability
5. Reliability

What is the use of different OOPs concepts in POM Model?

1. **Abstraction:** we write locators (such as id, name, xpath etc.,) in a Page Class. We utilize these locators in tests, but we can't see these locators in the tests.
2. **Interface:** WebDriver driver = new FirefoxDriver(); WebDriver itself is an Interface.
3. **Inheritance:** Base Class to initialize WebDriver interface, WebDriver waits, Property files, Excels, etc., in the Base Class. We extend the Base Class in other classes such as Tests and Utility Class.
4. **Encapsulation:** we declare the data members using @FindBy and initialization of data members will be done using Constructor to utilize those in methods.
5. **Method Overloading:** Implicit Wait, Action class in TestNG is also an example of overloading.
6. **Method Overriding:** Declaring a method in child class which is already present in the parent class is called Method Overriding. Examples are get and navigate methods of different drivers in Selenium.

What is Maven Lifecycle?

1. **Validate:** This step validates if the project structure is correct. For example – It checks if all the dependencies have been downloaded and are available in the local repository.
2. **Compile:** It compiles the source code, converts the .java files to .class and stores the classes in target/classes folder.
3. **Test:** It runs unit tests for the project.
4. **Package:** This step packages the compiled code in distributable format like JAR or WAR.
5. **Integration test:** It runs the integration tests for the project.
6. **Verify:** This step runs checks to verify that the project is valid and meets the quality standards.
7. **Install:** This step installs the packaged code to the local Maven repository.
8. **Deploy:** It copies the packaged code to the remote repository for sharing it with other developers.

What is Behaviour-Driven Development (BDD)?

1. BDD is a way for software teams to work that closes the gap between business people and technical people.

What is Cucumber?

1. Cucumber is a tool that supports Behaviour-Driven Development (BDD).

When is Cucumber used in real-time?

1. Cucumber tool is generally used in real-time to write acceptance tests for an application.

What is Gherkin?

1. Gherkin is a set of grammar rules that makes plain text structured enough for Cucumber to understand. The scenario above is written in Gherkin.
2. **Gherkin Language/Annotations in Cucumber:** Feature, Scenario, Given, When, Then, But, And, Background, etc.

What is the file extension of all the Gherkin files?

1. .feature

In Cucumber, comments are denoted by

1. #

Enlist the files needed in the Cucumber framework.

1. **Feature File (.feature):** It has plain text descriptions of single or numerous test situations.
2. **Step Definition File (.java)**: It has the extension .java. It essentially acts as a translator between the test scenario steps provided in the feature file and the automation code.
3. **TestRunner (.java):** .java is the file extension for this file. It connects the feature file and the step definition file. No code should be written under the TestRunner class.
    a. @RunWith(Cucumber.class)
    b. @CucumberOptions(features="Features",glue={"StepDefinition"})
    c. public class Runner {}

In a feature file, what is the maximum number of scenarios?

1. A feature file in Cucumber can include a maximum of 10 scenarios.

How to run multiple feature files using the Cucumber runner class?

1. features="src/test/resources/FeatureFiles",tags="@feature1 scenariogroup1,@feature2cenariogroup2"

How can Cucumber be integrated with Selenium WebDriver?

1. cucumber-core-1.2.2.jar
2. cucumber-java-1.2.2.jar
3. cucumber-junit-1.2.2.jar
4. cucumber-jvm-deps-1.0.3.jar
5. cucumber-reporting-0.1.0.jar
6. gherkin-2.12.2.jar

What are the Main Options available in Cucumber?

1.

Options Type	Purpose	Default Value
dryRun	true: Checks if all the Steps have the Step Definition	false
features	set: The paths of the feature files	{}
glue	set: The paths of the step definition files	{}
tags	instruct: What tags in the features files should be executed	{}
monochrome	true: Display the console Output in much readable way	false
format	set: What all report formaters to use	false
strict	true: Will fail execution if there are undefined or pending steps	false

What is the name of the plugin that is used to integrate Eclipse with Cucumber?

1. Cucumber Natural Plugin

What is the difference between Scenario and Scenario Outline?

1. The scenario outline is exactly similar to the scenario structure, but the only difference is the provision of multiple inputs.

What is the difference between Scenario Outline & Data Table?

1. Scenario Outline uses the Example Keyword to define the test data for the Scenario. This works for the whole test.
    a. Examples:
    b. | username | password |
    c. | testuser_1 | Test@153 |
2. Data Table does not use the Example Keyword. This works only for a single step.
    a. And User enters Credentials to LogIn
    b. | testuser_1 | Test@153 |

What are Maps in Data Tables?

1. Maps in Data Tables can be used in different ways. Headers can also be defined for the data tables. The same step can be executed multiple times with a different set of test data using Maps.

What are the tags in Cucumber?

1. In Cucumber, tags are used to associate a test like smoke, regression etc. with a particular scenario.
    a. @SmokeTest
    b. Scenario: Search contact
    c. Given: The desired contact will be displayed
2. Test Runner File: tags = {"@SmokeTest"}) // RUN all SmokeTest Cases.

How to ignore the test cases of a particular tag in Cucumber?

1. tags = {"~@SmokeTest"} // ignores the SmokeTest Cases

What are hooks in Cucumber?

1. Hooks are code blocks that execute before or after each Cucumber scenario in the execution cycle. Hooks are defined in the step definition file.

What is the Background Keyword in Cucumber?

1. Background in Cucumber is used to define a step or series of steps that are common to all the tests in the feature file.
2. It is defined in the feature file.

What is a Web Service?

1. It is a service offered by an electronic device to another electronic device, communicating with each other via the World Wide Web. Or
2. It is a server running on a computer device, listening for requests at a particular port over a network, serving web documents (HTML, JSON, XML, images).

What are the types of API Testing?

1. Unit Testing
2. Functional Testing
3. Load Testing
4. Security Testing
5. Penetration Testing

What are the protocols used in API Testing?

1. HTTP, REST, SOAP, JMS, etc.

What is the difference between REST and SOAP?

1. REST and SOAP are 2 different **approaches to online data transmission.**
2. Specifically, both define how to build application programming interfaces (APIs), which allow data to be communicated between web applications.
3. The main difference is that **SOAP is a protocol while REST is not.**
4. **Representational state transfer (REST)**
    a. It is a set of architectural principles.
    b. Because it's a set of guidelines, it leaves the implementation of these recommendations to developers.
5. **Simple object access protocol (SOAP)**
    a. It is an official protocol maintained by the World Wide Web Consortium (W3C).
    b. Because it is a protocol, it imposes built-in rules that increase its complexity and overhead, which can lead to longer page load times.

What exactly needs to verify in API testing?

1. We will verify the accuracy of the data.
2. Will see the HTTP status code.
3. We will see the response time.
4. Error codes in case API returns any errors.
5. Authorization would be check.
6. Non-Functional testing such as performance testing, security testing.

What is Postman?

1. We will verify the accuracy of the data.
2. Will see the HTTP status code.
3. We will see the response time.
4. Error codes in case API returns any errors.
5. Authorization would be check.
6. Non-Functional testing such as performance testing, security testing.

Why Postman?

1. Free
2. Easy
3. APIs Support
4. Extensible

What is a collection in Postman?

1. A collection in Postman helps to group similar requests. It helps in systematically arranging the requests into folders.

How can we stop executing requests or stop the collection run?

1. postman.setNextRequest(null);

What are Variables in Postman?

1. Variables enable you to store and reuse values in Postman.

How will you log variable values in Postman?

1. console.log(pm.variables.get("variable_name"));

How do you access postman variables?

1. {{variable_name}}

What are the different types of API requests supported in Postman?

1. GET, POST, PUT, PATCH, DELETE, COPY, HEAD, VIEW, TRACE, etc.

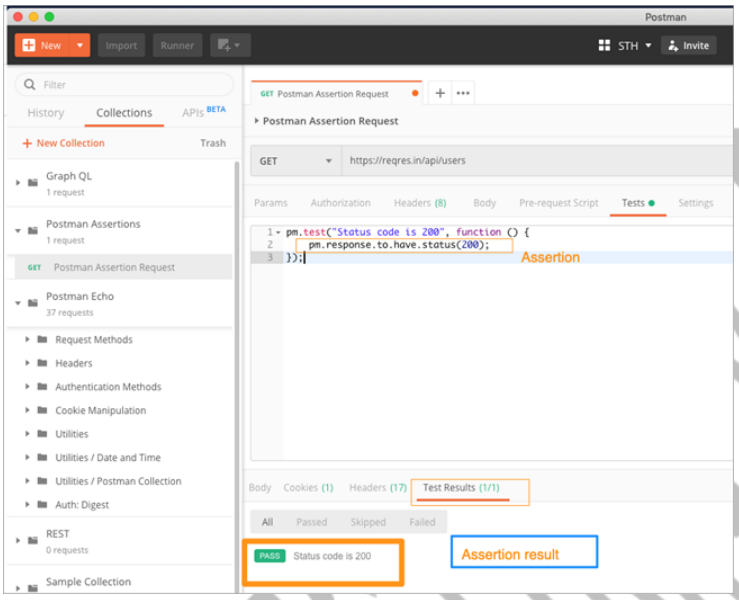
HTTP response status codes:

1. **Informational responses (100–199)**
    a. 100 Continue
    b. 101 Switching Protocols
    c. 102 Processing
2. **Successful responses (200–299)**
    a. 200 OK
    b. 201 Created
    c. 202 Accepted
    d. 204 No Content
3. **Redirection messages (300–399)**
    a. 300 Multiple Choice
    b. 301 Moved Permanently
4. **Client error responses (400–499)**
    a. 400 Bad Request
    b. 401 Unauthorized
    c. 402 Payment Required
    d. 403 Forbidden
    e. 404 Not Found
    f. 408 Request Timeout
5. **Server error responses (500–599)**
    a. 500 Internal Server Error
    b. 501 Not Implemented
    c. 502 Bad Gateway
    d. 503 Service Unavailable
    e. 504 Gateway Timeout

How to write Test Case in Postman? OR How response codes are validated in Postman?

It can be done using assertion in Postman.

1. pm.test("Status code is 200", function () {
2. pm.response.to.have.status(200);
3. });
    a. **pm.test()** is a closure function that allows you to write tests for a Postman request that's being executed.
    b. **pm.response.to.have.status(200)** is the actual assertion that is trying to validate the response to have a status code of 200.



What are the various authorization methods provided by Postman?

1. APIs use authorization to ensure that client requests access data securely.
2. Postman won't send authorization details with a request unless you specify an auth type.
    a. **API Key:** With API key auth, you send a key-value pair to the API either in the request headers or query parameters.
    b. **Bearer Token:** enable requests to authenticate using an access key, such as a JSON Web Token (JWT).
    c. **Basic Auth:** involves sending a verified username and password with your request.
    d. **Digest Auth:** the client sends a first request to the API, and the server responds with a few details, including a number that can be used only once (a nonce), a realm value, and a 401 unauthorized response.
    e. **AWS Signature:** AWS uses a custom HTTP scheme based on a keyed-HMAC (Hash Message Authentication Code) for authentication.
    f. **Oauth 1.0:** enables client applications to access data provided by a third-party API.
    g. **Oauth 2.0:** you first retrieve an access token for the API, then use that token to authenticate future requests.

What is Client URL (cURL)?

1. cURL, which stands for client URL, is a command line tool that developers use to transfer data to and from a server.
2. It supports about 22 protocols, including HTTP.
3. This combination makes it a very good ad-hoc tool for testing our REST services.

What is the Web Services Description Language (WSDL)?

1. It is an XML-based interface description language that is used for describing the functionality offered by a web service.
2. It provides a machine-readable description of how the service can be called, what parameters it expects, and what data structures it returns.

What is Appium?

1. Appium is an open-source test automation framework for use with native, hybrid and mobile web apps.

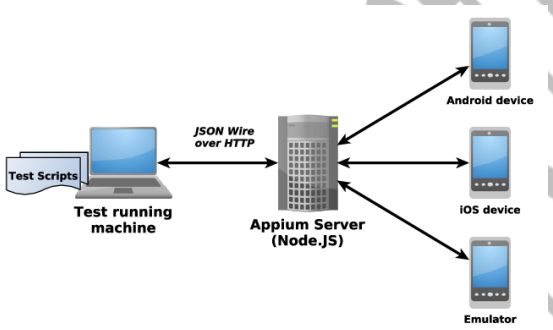
What is the difference between Emulator and Simulator?

1. Simulator: A simulator is used to simulate an environment with all the software variables and configurations that will be present in the actual production environment of an application.
2. Emulator: An emulator does try to replicate all the hardware and software aspects of a real-world environment. Emulators replicate both hardware and software features.

List Thing Which You Cannot Do with Emulators But You Can Do With A Real Device.

1. Phone calls & Messages
2. Battery drains out while using the application under test
3. Low battery scenarios
4. Memory card mount/unmount scenarios
5. Actual performance of your application
6. Bluetooth related testing.

Explain the architecture of Appium.



1.

Explain the JSON Wire protocol used by Appium.

1. The JSON Wire Protocol is the method by which client and server data are exchanged. It was created by WebDriver's developers.

What are the basic requirements for writing Appium tests?

1. Driver Client
2. Appiun Session
3. Desired Capabilities
4. Driver Commands

Can you explain the general structure of mobile application testing framework?

1. Application Package
2. Instrumentation Test Runner
3. Test Package

List out the prerequisites required to run tests on an android application in Appium locally.

1. JDK
2. Android Studio
3. Android SDK Tools
4. Eclipse IDE
5. Appium Desktop Client
6. TestNG
7. Selenium Server Jar

What are the main criteria under consideration when performing End to End Testing?

1. Installation ==> Launch Activity ==> UI Orientation ==> Network Activity ==> Reponse to various Devices ==> Performance Testing

List out the various application extensions along with their full forms:

1. **iOS:** IPA (iOS App Store Package)
2. **Android:** APK (Android Application Package File)

The minimum set of required capabilities for any Appium driver should include:

1. On iOS and Android, Appium behaves differently. Because it is a "cross-platform" tool, a method must be in place to distinguish between the session requests of the two operating systems.
2. JSON objects, referred to as Desired Capabilities, were introduced to address this specific problem statement.
    a. **platformName:** the name of the platform to automate.
    b. **platformVersion:** the version of the platform to automate.
    c. **deviceName:** the kind of device to automate.
    d. **app:** the path to the app you want to automate (but use the browserName capability instead in the case of automating a web browser).
    e. **app package:** the identifier of the application package.
    f. **app activity:** the name of the main application package.
    g. **automationName**: the name of the driver you wish to use.

How to initialize Appium Driver?

1. AppiumDriver driver=new AndroidDriver(new URL("http://127.0.0.1:4723/wd/hub"),capability);

How to automatically launch Android Emulator?

1. Caps.setCapability(capabilityName: “avd”, value: “Pixel_3”); // get the avd name from Android Studio
2. Caps.setCapability(“avdLaunchTimeout”, 180000);

What is the Maven Dependency for appium?

1. <dependency>
2. <groupId>io.appium</groupId>
3. <artifactId>java-client</artifactId>
4. <version>3.4.1</version>
5. </dependency>

Which is the vendor provided frameworks we use in Appium for Android and iOS devices?

1. Android 4.2+: Google's UIAutomator
2. iOS 9.3 and above: Apple's XCUITest.

How to Tap and Long Press on an element using Appium?

1. **Tap**
    a. TouchAction t =new TouchAction(driver);
    WebElement expandList= driver.findElementByXPath("//android.widget.TextView[@text='Expandable Lists']");
    t.tap(tapOptions().withElement(element(expandList))).perform();
    driver.findElementByXPath("//android.widget.TextView[@text='1. Custom Adapter']").click();
2. **Long_Press**
    a. WebElement pn= driver.findElementByXPath("//android.widget.TextView[@text='People Names']");
    t.longPress(longPressOptions().withElement(element(pn)).withDuration(ofSeconds(2))).release().perform();

How to Swipe using Appium?

1. TouchAction t=new TouchAction(driver); //long press //on element// 2 sec// move to another element and you release
    WebElement first=driver.findElementByXPath("//*[@content-desc='15']"); WebElement second=driver.findElementByXPath("//*[@content-desc='45']");
    t.longPress(longPressOptions().withElement(element(first)).withDuration(ofSeconds(2))).moveTo(element(second)).release().perform();

How to scroll using Appium?

1. driver.findElementByAndroidUIAutomator("new UiScrollable(new UiSelector()).scrollIntoView(text(\"WebView\"));");

How to drag and drop using Appium?

1. TouchAction action = new TouchAction(driver);
2. action.longPress(source).waitAction(3000).moveTo(destination).perform().release();

How to verify toast messages for error validations using Appium?

1. String toastMessage = driver.findElement
2. (By.xpath("//android.widget.Toast[1]")).getAttribute("name");
3. System.out.println(toastMessage);

How to dismiss the keyboard in Appium?

1. driver.hideKeyboard()

How to lock and unlock Android device using Appium?

1. ((AndroidDriver) driver).lockDevice();
2. System.out.println(((AndroidDriver) driver).isDeviceLocked());
3. ((AndroidDriver) driver).unlockDevice();

What are the various step for Automating hybrid apps using Appium?

1. Navigate to a portion of your app where a web view is active.
2. Retrieve the currently available contexts: This returns a list of contexts we can access, like **'NATIVE_APP' or 'WEBVIEW_1'.**
3. Set the context with the id of the context you want to access.
4. To stop automating in the web view context and go back to automating the native portion of the app, simply set the context again with the native context id (generally 'NATIVE_APP') to leave the web context and once again access the native commands.
5. // assuming we have a set of capabilities
6. driver = new AppiumDriver(new URL("http://127.0.0.1:4723/wd/hub"), capabilities);
7. Set<String> contextNames = driver.getContextHandles();
8. for (String contextName : contextNames) {
9. System.out.println(contextName); //prints out something like NATIVE_APP \n WEBVIEW_1
10. }
11. driver.context(contextNames.toArray()[1]); // set context to WEBVIEW_1
12. //do some web testing
13. String myText = driver.findElement(By.cssSelector(".green_button")).click();
14. driver.context("NATIVE_APP");
15. // do more native testing if we want
16. driver.quit();

How To: Export a crash log (logcat) from an Android device.

1. adb logcat >/$*some-folder*/android-debug.log