L1 Questions and answers:

1. **What is access specifier**

1. Public 2. Protected 3. Default 4. Private

1. Public:

➢ It is global level access(same package + different package)

2. Private:

➢ It is a class level access

3. Default:

➢ Package level access

➢ Without access specifier within the package we can access

4. Protected:

Inside package + outside Package (Extends)

2. **What is overloading and overriding with sample code**

Method Overloading

➢ In a same class method name is same and the argument is different is called method overloading

Eg:

```
public class StudentInfo {
private void studentId(int num) { }
private void studentId(String name) { \\ depends on order }
 private void studentId(String email, int ph) { \\depends on data type }
 private void studentId(int dob, String add) { \\depends on datatype count }
public static void main(String[] arg) {
StudentInfo info = new StudentInfo(); } }
```

Method Overriding

➢ In a different class, the method name should be same and argument name should be same is called overriding

Eg:

```
public class Employee extends Person {
public void personName () {
System.out.println("Preethi");
}
```

```
     public static void main(String[] args) {
                    Employee b=new Employee ();
                    b. personName ();
     }
```

➢ 2nd class(super class)

```
public class Person {
          public void personName() {
          System.out.println("priya"); }
```

output : Preethi;

3. Abstract and Interface

Abstract class:

➢ It is partially abstraction

➢ It support both abstract method and non-abstract method

➢ It’s using "extends" keyword

➢ Here "public abstract" have to mention

➢ We can use whatever access specifier we want

Example Program

```
public abstract class Bank {
abstract void saving(); //method signature
abstract void current();
abstract void salary();
 abstract void joint();
public void branchDetails(){
          System.out.println("chennai"); } }
```

super class

```
public class AxisBank extends Bank {
public void saving() { // method signature
 System.out.println("saving regular"); // business logic }
public void current() {
          System.out.println("current"); }
          public void salary() {
          System.out.println("salary"); }
          public void joint() {
```

```
    System.out.println("joint"); }
    public static void main(String[] args) {
AxisBank info = new AxisBank();
info.branchDetails();
info.salary();
info.saving(); }}
```

Output:

```
chennai
salary
saving regular
```

Interface:

- It is fully abstraction
- It support only abstract method
- It’s using "implement" keyword
- "public Abstract" is default. no need to mention
- Here we use only public(access specifier)

Example Program

```
public interface Bank {
abstract void saving();
abstract void current();
abstract void salary();
abstract void joint();
public void branchDetails(); }
super class
public class AxisBank implements Bank {
public void saving() {
        System.out.println("saving regular"); }
public void current() {
    System.out.println("current"); }
    public void salary() {
    System.out.println("salary"); }
    public void joint() {
    System.out.println("joint"); }
```

```
        public void branchDetails() {
        System.out.println("chennai"); } public static void main(String[] args) {
AxisBank info = new AxisBank(); info.branchDetails(); info.salary(); info.saving(); } }
```

4. What is hashmap, sample code to insert the values

   Hashmap:

   - It is key and value pair
   - It is a random order(based on key)

   Example Program:

```
public class ArList {
public static void main(String[] args) {
Map<Integer, String> ex = new HashMap<Integer, String>();
 ex.put(10, "Java");
ex.put(20, "Java");
ex.put(30, "sql");
ex.put(40, ".net");
ex.put(50, "sales");
ex.put(50, "fire");
 System.out.println(ex); } }
```

5. **What are the locators in selenium**

   Id,name, classname,xpath,tagname,linktext,partiallinktext and cssselector

6. **What is absolute and realative xpath**

   Absolute path

   - / represents absolute path
   - Absolute path (shows Full HTML DOM structure)

   Relative path(

   - // represents relative path
   - Relative path(shows particular locators)
   - ➢ In finding Xpath, matching node should be only one

7. **Dropdown SelectbyVisibleText sample code**

```
WebElement w = driver.findElement(By.name("coffee"));
Select s=new Select(w);
s.selectByVisibleText("With cream & sugar");
```

8. **If I click the button one alert is open how to switch over alert, then need to click OK , cancel and getText**

   ```
   driver.findElement(By.xpath("//input[@type='submit']")).click();
   Alert a = driver.switchTo().alert();
   System.out.println(a.getText());
   a.accept();
   a.dismiss();
    driver.switchTo().defaultContent();
   ```

9. **Do you know synchronization concepts**

   Yes, In Selenium we have implicit Wait and Explicit Wait conditional statements.

10. **Syntax for implicit and explicit wait**

    Implicit Wait Syntax:

    ```
    driver.manage().timeouts().implicitlyWait(time, TimeUnit.SECONDS);
    ```

Ex:

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

Explicit wait Syntax:

```
WebDriverWait wait = new WebDriverWait(driver,time);
```

Ex:

```
wait = new WebDriverWait(driver, time);
wait.until(ExpectedConditions.visibilityOf(element));
```

L2 Questions and answers:

1. **Syntax for implicit and explicit wait**

   Implicit Wait Syntax:

   ```
   driver.manage().timeouts().implicitlyWait(time, TimeUnit.SECONDS);
   ```

Ex:

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

Explicit wait Syntax:

```
WebDriverWait wait = new WebDriverWait(driver,time);
```

Ex:

```
wait = new WebDriverWait(driver, time);
wait.until(ExpectedConditions.visibilityOf(element));
```

2. How to run one group of tests

- In TestNG we can execute only set of groups
- In test method we should mention like this Test(groups={“facebook”})
- Groups are specified in testng.xml file .

3. What is Data Provider

- parameters can be passed using Dataproviders.
- A Data Provider is a method annotated with *@DataProvider*.
- A Data Provider returns an array of objects.
- Ex: @Test(dataProvider="getData")

4. I have click the button then I write it for switch to alert but alert is not present how do u handle

- We can handle it by using try catch block
- Then we can wait until alert is present by using alertIsPresent(), then we can switch to alert
- **Ex:**

```
          public void checkAlert() {
             try {
           WebDriverWait wait = new WebDriverWait(driver, 2);
          wait.until(ExpectedConditions.alertIsPresent());
           Alert alert = driver.switchTo().alert();
           alert.accept();
   } catch (Exception e) {
        //exception handling
   }
}
```

5. Write code for windows handling

```
String parentWindowId = driver.getWindowHandle();
System.out.println("Parent Window ID:" + parentWindowId);
driver.findElement(By.id("loginsubmit")).click();
Set<String> allWindowId = driver.getWindowHandles();
Iterator<String> a = allWindowId.iterator();
while (a.hasNext()) {
String s = a.next();
System.out.println("All Windows ID:" + s);
if (!parentWindowId.endsWith(s)) {
System.out.println("Child Window ID:" + s);
driver.switchTo().window(s);
```

6. Mouse Over action code

Sample Code:

```
WebElement web = driver.findElement(By.xpath(".//*[text()='Product Category']"));
Actions a=new Actions(driver);
a.moveToElement(web).perform();
```

7. Write the code for read the data from excel

```
List<HashMap<String, String>> mapDatasList = new ArrayList(); try {
File excelLocaltion = new File("./Excel/Adactin.xlsx");
String sheetName = "Adact";
FileInputStream f = new FileInputStream(
        excelLocaltion.getAbsolutePath());
        Workbook w = new XSSFWorkbook(f);
        Sheet sheet = w.getSheet(sheetName);
        Row headerRow = sheet.getRow(0);
        for (int i = 0; i < sheet.getPhysicalNumberOfRows(); i++) {
    Row currentRow = sheet.getRow(i);
    HashMap<String, String> mapDatas = new HashMap<String, String>();
    for (int j = 0; j < headerRow.getPhysicalNumberOfCells(); j++) {
Cell currentCell = currentRow.getCell(j);
switch (currentCell.getCellType()) { case Cell.CELL_TYPE_STRING:
mapDatas.put(headerRow.getCell(j).getStringCellValue(),
        currentCell.getStringCellValue()); break; case Cell.CELL_TYPE_NUMERIC:
mapDatas.put(headerRow.getCell(j).getStringCellValue(),
String.valueOf(currentCell
.getNumericCellValue()));
break;
    } }
    mapDatasList.add(mapDatas); }
// System.out.println(mapDatasList);
String s = mapDatasList.get(1).get("Username"); String s1 =
mapDatasList.get(1).get("Password"); System.out.println(s); System.out.println(s1);
} catch (Throwable e) {
    e.printStackTrace();
```