

Important topics for interviews(only written):

COREJAVA:

1.Inheritance(Single, Multilevel):

Single Inheritance:

```
public class A {
    public void test1() {
        System.out.println("test1");
    }
}

public class B extends A {
    public void test2() {
        System.out.println("test2");
    }

    public static void main(String[] args) {
        B b1 = new B();
        b1.test1();
        b1.test2();
    }
}
```

Multilevel Inheritance:

```
public class C {
    public void test3() {
        System.out.println("test3");
    }
}

public class A extends C {
    public void test1() {
        System.out.println("test1");
    }
}

public class B extends A {
    public void test2() {
        System.out.println("test2");
    }

    public static void main(String[] args) {
        B b1 = new B();
        b1.test1();
        b1.test2();
        b1.test3();
    }
}
```

2.Polymorphism(Method overloading,Method overriding):

Method overloading:

```

public class Employee {
    public void findEmpId(int age) {
        System.out.println("123");
    }

    public void findEmpId() {
        System.out.println("123");
    }

    public static void main(String[] args) {
        Employee e = new Employee();
        e.findEmpId();
        e.findEmpId(765);
    }
}

```

Method overriding:

```

public class AxisBank {
    public void fixedDeposit() {
        System.out.println("5%");
    }
}
public class HDFCBank extends AxisBank {
    public void fixedDeposit() {
        System.out.println("10%");
    }

    public static void main(String[] args) {
        AxisBank bank = new HDFCBank();
        bank.fixedDeposit();
    }
}

```

3. Abstractions (interface, abstract class, multiple inheritance through interface):

Abstract class:

```

public abstract class Bank {
    public abstract void savingAcct();

    public abstract void currentAcct();

    public void branchDetails() {
        System.out.println("Chennai");
    }
}

public class HDFCBank extends Bank {

    public void savingAcct() {
        System.out.println("6%");
    }

}

```

```

public void currentAcct() {
    System.out.println("8%");
}

public static void main(String[] args) {
    Bank b = new HDFCBank();
    b.savingAcct();
    b.currentAcct();
    b.branchDetails();
}
}

```

```

public class AxisBank extends Bank {

    public void savingAcct() {
        System.out.println("5%");
    }

    public void currentAcct() {
        System.out.println("10%");
    }

    public static void main(String[] args) {
        Bank b = new AxisBank();
        b.savingAcct();
        b.currentAcct();
        b.branchDetails();
    }
}

```

Interface:

```

public interface Bank {
    public abstract void savingAcct();

    public void currentAcct();
}

```

```

public class HDFCBank implements Bank {

    public void savingAcct() {
        System.out.println("6%");
    }

    public void currentAcct() {
        System.out.println("8%");
    }

    public static void main(String[] args) {
        Bank b = new HDFCBank();
        b.savingAcct();
        b.currentAcct();
    }
}

```

```

    }
}

public class AxisBank implements Bank {

    public void savingAcct() {
        System.out.println("5%");
    }

    public void currentAcct() {
        System.out.println("10%");
    }

    public static void main(String[] args) {
        Bank b = new AxisBank();
        b.savingAcct();
        b.currentAcct();
    }
}

```

Multiple Inheritance through Interface:

```

public interface I1 {
    void test1();
}

public interface I2 {
    void test2();
}

public class C implements I1, I2 {

    public void test2() {
        System.out.println("test2");
    }

    public void test1() {
        System.out.println("test1");
    }

    public static void main(String[] args) {
        C c1 = new C();
        c1.test1();
        c1.test2();
    }
}

```

4. Insert values in list/userdefine and iterate values using foreach/for:

Insert values in List:

```

import java.util.ArrayList;
import java.util.List;

public class Employee {
    public static void main(String[] args) {
        List<Integer> emp = new ArrayList<>();
        emp.add(12);
        emp.add(122);
        emp.add(123);
        emp.add(124);
        emp.add(125);

        // for loop

        for (int i = 0; i < emp.size(); i++) {
            System.out.println(emp.get(i));
        }

        // Enhanced for loop
        for (Integer i : emp) {
            System.out.println(i);
        }
    }
}

```

Insert values in List with userdefine list:

```

public class Employee {
    private int id;
    private String name;

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }
}

```

```

import java.util.ArrayList;
import java.util.List;

public class Details {
    public static void main(String[] args) {
        List<Employee> emp = new ArrayList<>();
    }
}

```

```

Employee e1 = new Employee();
e1.setId(12);
e1.setName("Vel");

Employee e2 = new Employee();
e2.setId(13);
e2.setName("Bala");

emp.add(e1);
emp.add(e2);

for (Employee x : emp) {
    System.out.println(x.getId());
    System.out.println(x.getName());
}
}
}

```

5. Insert values in Map/userdefine Map and iterate values:

Insert the values in Map:

```
package org.cts.toyota.login;
```

```
import java.util.HashMap;
import java.util.Map;
import java.util.Map.Entry;
import java.util.Set;
```

```
public class Details {
    public static void main(String[] args) {

        Map<Integer, String> emp = new HashMap<>();
        emp.put(10, "Selenium");
        emp.put(20, "Webdriver");
        emp.put(30, "JUnit");
        emp.put(40, "TestNG");
        emp.put(50, "SQL");
        Set<Entry<Integer, String>> e = emp.entrySet();
        for (Entry<Integer, String> x : e) {
            System.out.println(x.getKey());
            System.out.println(x.getValue());
        }
    }
}

```

Insert the values in userdefine Map:

```
public class Employee {
    private int id;
    private String name;

    public int getId() {
        return id;
    }

    public void setId(int id) {

```

```

        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }
}

import java.util.HashMap;
import java.util.Map;
import java.util.Map.Entry;
import java.util.Set;

public class Details {
    public static void main(String[] args) {

        Map<Integer, Employee> emp = new HashMap<>();

        Employee e1 = new Employee();
        e1.setId(12);
        e1.setName("bala");

        Employee e2 = new Employee();
        e2.setId(312);
        e2.setName("Guna");

        emp.put(10, e1);
        emp.put(20, e2);

        Set<Entry<Integer, Employee>> e = emp.entrySet();
        for (Entry<Integer, Employee> x : e) {
            System.out.println(x.getKey());
            System.out.println(x.getValue().getId());
            System.out.println(x.getValue().getName());
        }

    }
}

```

6.JDBC connection steps code:

JDBC connection steps:

```

package org.cts.toyota.login;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

```

```

public class Details {
    public static void main(String[] args) {
        Connection con = null;
        try {
            // load the driver
            Class.forName("com.mysql.jdbc.Driver");
            // connec to db
            con =
DriverManager.getConnection("jdbc:mysql://127.0.0.1localhost:portnum/schema
Name", "username",
            "password");
            // write a sql query
            String sql = "Select * from employees";
            PreparedStatement ps = con.prepareStatement(sql);
            // execute query
            ResultSet rs = ps.executeQuery();
            // iterate
            while (rs.next()) {
                int empId = rs.getInt("id");
                String empName = rs.getString("name");
            }

        } catch (Throwable e) {
            e.printStackTrace();
        } finally {
            try {
                // close db connection
                con.close();
            } catch (SQLException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }
}

```

7.Code for singleton class:

```

public class Employee {
    private static Employee emp = null;

    private Employee() {
    }

    public static Employee getObject() {
        if (emp == null) {
            emp = new Employee();
        }
        return emp;
    }

    public void getId() {
        System.out.println("12");
    }

    public void getName() {
        System.out.println("Vel");
    }
}

```



```

public class Details {
    public static void main(String[] args) {
        Employee e = Employee.getObject();
        e.getId();
        e.getName();

    }
}

```

8.userdefine exception:

```

public class MyException extends Exception {
    @Override
    public String getMessage() {
        return "Not Found";
    }
}

```

```

public class Details {
    public static void main(String[] args) {
        String env = "down";
        if (env.equals("down")) {
            try {
                throw new MyException();
            } catch (MyException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    }
}

```

Windows Handling

Syntax:

```

driver.switchTo().window(String Url)
driver.switchTo().window(String title)
driver.switchTo().window(String windows)

```

```

driver.getWindowHandle()-parent
driver.getWindowHandles()--child

```

```

Set<String> all=driver.getWindowsHandles();
List<String> emp=new ArrayList<String>();
emp.addAll(all);
driver.switchTo().window(emp.get(index));

```

Program:

```

import java.util.Set;

```

```

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

```

```

public class Employee {

```

```

public static void main(String[] args) {
    System.setProperty("webdriver.chrome.driver", "path");
    WebDriver driver = new ChromeDriver();
    driver.get("url");
    // parent window id
    String pWid = driver.getWindowHandle();
    // all windows
    Set<String> allWindows = driver.getWindowHandles();
    for (String x : allWindows) {
        if (!pWid.equals(x)) {
            driver.switchTo().window(x);
        }
    }
}
}

```

webtable

Syntax:

```

List<WebElement> tRows=driver.findElements(By.tagName("tr"));
for(WebElement x:tRows){
List<WebElement> tData=driver.findElements(By.tagName("td"));
for(WebElement y:tData){
String name=y.getText();
sysout(name);
}
}

```

Program:

```

package org.cts.toyota.login;

```

```

import java.util.List;

```

```

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

```

```

public class Employee {
    public static void main(String[] args) {
        System.setProperty("webdriver.chrome.driver", "path");
        WebDriver driver = new ChromeDriver();
        driver.get("url");
        // Enhanced for loop
        List<WebElement> tRows = driver.findElements(By.tagName("tr"));
        for (WebElement rows : tRows) {
            List<WebElement> tData = rows.findElements(By.tagName("td"));
            for (WebElement data : tData) {
                System.out.println(data.getText());
            }
        }

        // for loop
        for (int i = 0; i < tRows.size(); i++) {
            List<WebElement> tData = tRows.get(i).findElements(By.tagName("td"));
            for (int j = 0; j < tData.size(); j++) {
                System.out.println(tData.get(j).getText());
            }
        }
    }
}

```

```
}
```

ScreenShot

Syntax:

```
TakeScreenShot tk=(TakeScreenShot)driver;  
File src=tk.getScreenShotAs(OutPut type.FILE);  
File desc=new File("path");  
FileUtiles.copyFile(Src,desc);
```

Program:

```
import java.io.File;  
import java.io.IOException;  
  
import org.apache.commons.io.FileUtils;  
import org.openqa.selenium.OutputType;  
import org.openqa.selenium.TakesScreenshot;  
import org.openqa.selenium.WebDriver;  
import org.openqa.selenium.chrome.ChromeDriver;
```

```
public class Employee {  
    public static void main(String[] args) throws IOException {  
        System.setProperty("webdriver.chrome.driver", "path");  
        WebDriver driver = new ChromeDriver();  
        driver.get("url");  
  
        TakesScreenshot tk = (TakesScreenshot) driver;  
        File f = tk.getScreenShotAs(OutputType.FILE);  
        FileUtils.copyFile(f, new File("dest location"));  
  
    }  
}
```

ScrollDown

Syntax:

```
JavaScriptExecutor jk=(JavaScriptExecutor)driver;  
jk.executeScript("arguments[0].ScrollIntoview(true)",WebElement)  
ScrollUp
```

```
JavaScriptExecutor jk=(JavaScriptExecutor)driver;  
jk.executeScript("arguments[0].ScrollIntoview(false)",WebElement)
```

Program:

```
package org.cts.toyota.login;  
  
import java.io.IOException;  
  
import org.openqa.selenium.By;  
import org.openqa.selenium.JavascriptExecutor;  
import org.openqa.selenium.WebDriver;  
import org.openqa.selenium.WebElement;  
import org.openqa.selenium.chrome.ChromeDriver;  
  
public class Employee {  
    public static void main(String[] args) throws IOException {  
        System.setProperty("webdriver.chrome.driver", "path");  
        WebDriver driver = new ChromeDriver();  
        driver.get("url");  
  
        WebElement down = driver.findElement(By.id("email"));  
        JavascriptExecutor js = (JavascriptExecutor) driver;  
        // down  
        js.executeScript("arguments[0].scrollIntoView(true)", down);  
        // up  
        js.executeScript("arguments[0].scrollIntoView(false)", down);  
    }  
}
```

```
}  
}
```

link count

Program:

```
import java.io.IOException;  
import java.util.List;
```

```
import org.openqa.selenium.By;  
import org.openqa.selenium.WebDriver;  
import org.openqa.selenium.WebElement;  
import org.openqa.selenium.chrome.ChromeDriver;
```

```
public class Employee {  
    public static void main(String[] args) throws IOException {  
        System.setProperty("webdriver.chrome.driver", "path");  
        WebDriver driver = new ChromeDriver();  
        driver.get("url");  
        List<WebElement> allLinks = driver.findElements(By.tagName("a"));  
        int linksCount = allLinks.size();  
        System.out.println(linksCount);  
    }  
}
```

Broken link count

```
URLConnection connection = (URLConnection) url.openConnection();
```

Try

```
{  
connection.connect();  
String response = connection.getResponseMessage();  
connection.disconnect();  
return response;  
}  
catch (Exception exp)  
{  
return exp.getMessage();  
}  
}
```

Select

Syntax:

```
Select refName=new Select(webElement);  
refName.selectByIndex();  
refName.selectByVisibleText();  
refName.selectByValue();
```

Methods in Select:

```
selectByIndex();  
selectByVisibleText();  
selectByValue();  
getOptions();  
getAllSelectedOptions();  
getFirstSelectedOptions();  
isMultiple();  
deselectByIndex();  
deselectByVisibleText();  
deselectByValue();  
deselectAll();
```

For print the selected value:

```
import java.io.IOException;  
import java.util.List;
```

```

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.Select;

public class Employee {
    public static void main(String[] args) throws IOException {
        System.setProperty("webdriver.chrome.driver", "path");
        WebDriver driver = new ChromeDriver();
        driver.get("url");
        WebElement w = driver.findElement(By.id("country"));
        Select s = new Select(w);
        // foreach
        List<WebElement> emp = s.getAllSelectedOptions();
        for (WebElement x : emp) {
            System.out.println(x.getText());
        }
        // for loop
        for (int i = 0; i < emp.size(); i++) {
            System.out.println(emp.get(i).getAttribute("value"));
        }
    }
}

```

For print all the options:

```

package org.cts.toyota.login;

import java.io.IOException;
import java.util.List;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.Select;

public class Employee {
    public static void main(String[] args) throws IOException {
        System.setProperty("webdriver.chrome.driver", "path");
        WebDriver driver = new ChromeDriver();
        driver.get("url");
        WebElement w = driver.findElement(By.id("country"));
        Select s = new Select(w);
        // select all values by index
        List<WebElement> optionsAll = s.getOptions();
        for (int i = 0; i < optionsAll.size(); i++) {
            s.selectByIndex(i);
        }
        // select all values by value with for loop
        for (int i = 0; i < optionsAll.size(); i++) {
            s.selectByValue(optionsAll.get(i).getAttribute("value"));
        }
        // select all values by value with foreach
        for (WebElement x : optionsAll) {

```

```

        s.selectByValue(x.getAttribute("value"));
    }
    // select values by visibletext with for loop
    for (int i = 0; i < optionsAll.size(); i++) {
        s.selectByVisibleText(optionsAll.get(i).getText());
    }

    // select all values by visible text with foreach
    for (WebElement x : optionsAll) {
        s.selectByVisibleText(x.getText());
    }
}
}

```

Action:

mouseOverAction:

```

Actions refName=new Actions(driver);
refName.moveToElement(webElement).perform();

```

Program:

```

package org.cts.toyota.login;

```

```

import java.io.IOException;

```

```

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.interactions.Actions;

```

```

public class Employee {
    public static void main(String[] args) throws IOException {
        System.setProperty("webdriver.chrome.driver", "path");
        WebDriver driver = new ChromeDriver();
        driver.get("url");
        // drag and drop
        WebElement from = driver.findElement(By.id("from"));
        Actions acc = new Actions(driver);
        acc.moveToElement(from).perform();
    }
}

```

dropAndDrag:

```

Actions refName=new Actions(driver);
refName.dropAndDrag(webEle1,wenEle2).perform();

```

Program:

```

import java.io.IOException;

```

```

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.interactions.Actions;

```

```

public class Employee {
    public static void main(String[] args) throws IOException {
        System.setProperty("webdriver.chrome.driver", "path");
        WebDriver driver = new ChromeDriver();
    }
}

```

```

        driver.get("url");
        // drag and drop
        WebElement from = driver.findElement(By.id("from"));
        WebElement dest = driver.findElement(By.id("to"));
        Actions acc = new Actions(driver);
        acc.dragAndDrop(from, dest).perform();
    }
}

```

```

contextClick:
Actions refName=new Actions(driver);
refName.contextClick(webElement).perform();
Program:
import java.io.IOException;

```

```

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.interactions.Actions;

```

```

public class Employee {
    public static void main(String[] args) throws IOException {
        System.setProperty("webdriver.chrome.driver", "path");
        WebDriver driver = new ChromeDriver();
        driver.get("url");
        // drag and drop
        WebElement from = driver.findElement(By.id("from"));
        Actions acc = new Actions(driver);
        acc.contextClick(from).perform();
    }
}

```

```

DoubleClick:
Actions refName=new Actions(driver);
refName.doubleClick(webElement).perform();
Program:
import java.io.IOException;

```

```

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.interactions.Actions;

```

```

public class Employee {
    public static void main(String[] args) throws IOException {
        System.setProperty("webdriver.chrome.driver", "path");
        WebDriver driver = new ChromeDriver();
        driver.get("url");
        // drag and drop
        WebElement from = driver.findElement(By.id("from"));
        Actions acc = new Actions(driver);
        acc.doubleClick(from).perform();
    }
}

```

```

Implicit wait:
*It make the whole webdriver wait for that particular time we mention.
*driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS)
Explicit wait:

```

\*It make the particular webelement wait for that particular time we mention.

\*We can also mention the condition for which the webelement will wait.

```

*WebDriverWait wait = new WebDriverWait(driver, 10);
WebElement element =
wait.until(ExpectedConditions.somecondition(By.id("someid")));

```

Fluent wait:

\*Each FluentWait instance defines the maximum amount of time to wait for a condition, as well as the frequency with which to check the condition.

\*The user may configure the wait to ignore specific types of exceptions whilst waiting, such as NoSuchElementException when searching for an element on the page.

Xpath functions

Following:

\*Selects all elements in the document of the current node.

sibling:

\*Select the following siblings of the context node. Siblings are at the same level of the current node .

Parent:

\*Selects the parent of the current node.

Child :

\*Selects all children elements of the current node.

Preceding:

\*Select all nodes that come before the current node.

Alert

```
Alert refName=driver.SwitchTo.alert();
```

Program:

```
import java.io.IOException;
```

```
import org.openqa.selenium.Alert;
```

```
import org.openqa.selenium.WebDriver;
```

```
import org.openqa.selenium.chrome.ChromeDriver;
```

```

public class Employee {
    public static void main(String[] args) throws IOException {
        System.setProperty("webdriver.chrome.driver", "path");
        WebDriver driver = new ChromeDriver();
        driver.get("url");
        // alert switch
        Alert al = driver.switchTo().alert();
        // ok
        al.accept();
        // cancell
        al.dismiss();
        // insert
        al.sendKeys("yes");
    }
}

```

Frames:

Syntax:

```
driver.switchTo().frame("name");
```

```
driver.switchTo().frame(index);
```

```
driver.switchTo().frame(WebElement);
```

Program:

```
import java.io.IOException;
```

```
import java.util.List;
```

```
import org.openqa.selenium.By;
```



```

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class Employee {
    public static void main(String[] args) throws IOException {
        System.setProperty("webdriver.chrome.driver", "path");
        WebDriver driver = new ChromeDriver();
        driver.get("url");
        List<WebElement> emp = driver.findElements(By.tagName("iframe"));
        int frameCount = emp.size();
        System.out.println(frameCount);
    }
}

```

Reusable code for sendkeys:

```

public static void type(WebElement element,String name)
{
    element.sendKeys(name);
}

```

Reusable code for button:

```

public static void type(WebElement element)
{
    element.click();
}

```

Reusable code for driver close:

```

public static void type(WebDriver driver)
{
    driver.quit();
}

```

Reusable code for Driver lanuch:

```

public static webDriver getDriver(String url){
    System setProperty("webdriver.chrome.driver","path");
    WebDriver driver=new ChromeDriver();
    driver.get(url);
    return driver;
}

```