K.L.E. Society's
Raja Lakhamagouda Science Institute
(Autonomous), Belagavi
Bachelor of Computer Applications

5th Semester NEP 2022-23

Software Engineering & Testing lab Manual

1. **Write Test cases on the Login module of any website.**
   Refer Excel
2. **Write Defect report on any website.**
   Refer Excel
3. **Installation and configuration of Selenium WebDriver and Xampp.**

How to Download & Install Selenium WebDriver

Step 1 – Install Java Software Development Kit (JDK)

Download and install the **Java Software Development Kit (JDK).**

This JDK version comes bundled with Java Runtime Environment (JRE), so you do not need to download and install the JRE separately.

Once installation is complete, open command prompt and type "java" to check if java installation is successful

Step 2 – Install Eclipse IDE

Download the latest version of **"Eclipse IDE for Java Developers"**, Make sure to choose correctly between Windows 32 Bit and 64 Bit versions.

You should be able to download an exe file named "eclipse-inst-win64" for Setup.

Double-click on a file to install the Eclipse. A new window will open. Click Eclipse IDE for Java Developers.

After that, a new window will open which click button marked 1 and change path to "C:\eclipse". Post that Click on the Install button marked 2

After successful completion of the installation procedure, a window will appear. On that window click on Launch.

Step 3 – Selenium WebDriver Installation

You can download **Selenium Web driver for Java Client Driver**. You will find client drivers for other languages there, but only choose the one for Java.

This download comes as a ZIP file named "selenium-3.14.0.zip". For simplicity of Selenium installation on Windows 10, extract the contents of this ZIP file on your C drive so that you would have the directory "C:\selenium-3.14.0\". This directory contains all the JAR files that we would later import on Eclipse for Selenium setup.

Step 4 – Configure Eclipse IDE with WebDriver

1. Launch the "eclipse.exe" file inside the "eclipse" folder that we extracted in step 2. If you followed step 2 correctly, the executable should be located on C:\eclipse\eclipse.exe.
2. When asked to select for a workspace, just accept the default location.
3. 3. Create a new project through File > New > Java Project. Name the project as "new project".

A new pop-up window will open. Enter details as follow

Project Name

1. Location to save a project
2. Select an execution JRE
3. Select the layout project option
4. Click on the Finish button

4. In this step,

1. Right-click on the newly created project and
2. Select New > Package, and name that package as "new package".

A pop-up window will open to name the package,

1. Enter the name of the package
2. Click on the Finish button

5. Create a new Java class under newpackage by right-clicking on it and then selecting-New > Class, and then name it as "MyClass". Your Eclipse IDE

When you click on Class, a pop-up window will open, enter details as

1. Name of the class
2. Click on the Finish button

Now selenium WebDriver's into Java Build Path

6. In this step,
    1. Right-click on "newproject" and select **Properties**.

2. On the Properties dialog, click on “Java Build Path”.
3. Click on the **Libraries** tab, and then
4. Click on “Add External JARs.
5. When you click on “Add External JARs..” It will open a pop-up window. Select the JAR files you want to add.
6. After selecting jar files, click on OK button.
7. Select all files inside the lib folder.
8. Select files outside lib folder
9. Once done, click “Apply and Close” button.
10. Add all the JAR files inside and outside the “libs” folder. Your Properties dialog should now look similar to the image below.
11. Finally, click OK and we are done importing Selenium libraries into our project.

Installing XAMPP

Steps to install XAMPP on Windows:

- In the web browser,download XAMPP installer.

- During the installation process, select the required components like MySQL, FileZilla ftp server, PHP, phpMyAdmin or leave the default options and click the **Next** button.

- Uncheck the **Learn more about bitnami** option and click **Next** button.
- Choose the root directory path to set up the *htdocs* folder for our applications. For example *‘C:\xampp’.*
- Click the **Allow access** button to allow the XAMPP modules from the Windows firewall.
- After the installation process, click the **Finish** button of the XAMPP Setup wizard.
- Now the XAMPP icon is clearly visible on the right side of start menu. Show or Hide can be set by using the control panel by clicking on the icon.
- To start Apache and MySql, just click on the **Start** button on the control panel.

Note: Suppose Apache is not starting, it means some other service is running at port **80.** In this case, stop the other service temporarily and restart it.

4. Write Selenium script to demonstrate Selenium WebDriver basic commands

```
package com.test;

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

public class commands {

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

System.setProperty("webDriver.chrome.driver","F:\\Software Testing\\ST
Lab\\chromedriver.exe");
		WebDriver driver=new ChromeDriver();
		driver.get("https://www.google.com/");
		driver.manage().window().maximize();
		String title =driver.getTitle();
		System.out.println(title);
		Thread.sleep(1000);
		WebElement t=driver.findElement(By.tagName("textarea"));
		t.sendKeys("youtube");
		t.submit();
		driver.findElement(By.xpath("//h3[contains(text(),'YouTube')]")).click();
		driver.navigate().refresh();
		String url=driver.getCurrentUrl();
		System.out.println(url);
		driver.manage().window().fullscreen();
		driver.navigate().back();
		Dimension d=new Dimension(300,650);
		driver.manage().window().setSize(d);
		Thread.sleep(2000);
		driver.close();
		}
}
```

5. Write Selenium script to demonstrate Selenium WebDriver locators

```
package Automation;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class locators {

	public static void main(String[] args) throws InterruptedException {
		WebDriver driver=new ChromeDriver();
		driver.navigate().to("https://artoftesting.com/samplesiteforselenium");
		driver.manage().window().maximize();
		driver.manage().timeouts().implicitlyWait(5,TimeUnit.SECONDS);

		String sampleText=driver.findElement(By.id("idOfDiv")).getText();
	System.out.println(sampleText);
		Thread.sleep(3000);
	driver.findElement(By.linkText("This is a link")).click();
	driver.navigate().back();

	driver.findElement(By.name("firstName")).sendKeys("Hi");
	driver.findElement(By.name("firstName")).clear();

	driver.findElement(By.xpath("//button[@id='idOfButton']")).click();
	Thread.sleep(3000);
	driver.findElement(By.cssSelector("#female")).click();

	driver.findElement(By.className("Automation")).click();
	Thread.sleep(3000);
	WebElement image=driver.findElement(By.tagName("img"));
	System.out.println(image.getAttribute("alt"));

	driver.findElement(By.partialLinkText("Tutorial")).click();


	driver.quit();

	}

}
```

6. Write a script to demonstrate Gmail login using Selenium WebDriver

- **Launch the browser and open "Gmail.com".**
- **Verify the title of the page and print the verification result.**
- **Enter the username and Password.**
- **Click on the Sign in button.**
- **Wait for 10 seconds**
- **Close the web browser**

```
package Automation;

import java.time.Duration;
import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;

public class gmail {

    public static void main(String[] args) throws InterruptedException {
            WebDriver driver=new ChromeDriver();
      driver.manage().window().maximize();
            driver.navigate().to("https://gmail.com/");
            String expectedTitle="Gmail";
            String actualTitle=driver.getTitle();
            System.out.println(actualTitle);
            driver.manage().timeouts().implicitlyWait(5,TimeUnit.SECONDS);
            Thread.sleep(5000);
            if(actualTitle.equals(expectedTitle))
            {
                    System.out.println("Title is matching");
            }
            else {
                    System.out.println("Title is not matching");
            }

    driver.findElement(By.id("identifierId")).sendKeys("rlsbcademo2024@gmail.com");
            driver.findElement(By.xpath("//span[normalize-space()='Next']")).click();

            WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(30));

    wait.until(ExpectedConditions.elementToBeClickable(By.cssSelector("input[name='P
asswd']")));


    driver.findElement(By.cssSelector("input[name='Passwd']")).sendKeys("rlsbca123");

            driver.findElement(By.xpath("//span[normalize-space()='Next']")).click();
```

```
            Thread.sleep(3000);
            System.out.println("Gmail login successful");

            driver.close();

        }

    }
```

7. Write a script to add an item to cart of Amazon using Window handles method using Selenium WebDriver

```
package Selenium;

import java.util.ArrayList;

import java.util.concurrent.TimeUnit;

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

public class amazon {

        public static void main(String[] args) throws InterruptedException {
            WebDriver driver=new ChromeDriver();

            driver.manage().window().maximize();

            driver.navigate().to("https://www.amazon.in/");
            driver.manage().timeouts().implicitlyWait(5,TimeUnit.SECONDS);

            driver.manage().deleteAllCookies();

            WebElement
searchbox=driver.findElement(By.id("twotabsearchtextbox"));
            searchbox.sendKeys("iphone 14");
            searchbox.submit();

            driver.findElement(By.xpath("//span[@class='a-size-medium a-color-
base a-text-normal']")).click();

            // store window ids in array list
            ArrayList<String> wid = new
ArrayList<String>(driver.getWindowHandles());
```

```
                    //switch to active tab
                    driver.switchTo().window(wid.get(1));

                    driver.findElement(By.id("buy-now-button")).click();
                    Thread.sleep(3000);

                    System.out.println("Thank You for Shopping");

                    driver.quit();
                }
        }
```

8. Black Box testing: (Load Testing) Design a web page to get the count of visitors who visit your web page

```
<?php
// File to store the visitor count
$countFile = 'visitor_count.txt';
// Function to increment the count and retrieve the updated value
function incrementVisitorCount() {
global $countFile;
$count = 1;
if (file_exists($countFile)) {
$count = intval(file_get_contents($countFile));
$count++;
}
file_put_contents($countFile, $count);
return $count;
}
// Get the visitor count
$visitorCount = incrementVisitorCount();
?>
<!DOCTYPE html>
<html>
<head>
<title>Visitor Count Page</title>
<style>
```

```
body {
font-family: Arial, sans-serif;
text-align: center;
}
h1 {
margin-top: 50px;
}
#counter {
font-size: 48px;
margin-top: 20px;
}
</style>
</head>
<body>
<h1>Welcome to the Visitor Count Page</h1>
<p>Number of Visitors:</p>
<div id="counter"><?php echo $visitorCount; ?></div>
</body>
</html>
```

9. Black Box testing: (Functional Testing) Write a selenium script to test total number of objects present on the web page

```
package SeleniumWebDriver;
import java.util.List;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class noOfObjects {

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

```
                System.setProperty("webdriver.chrome.driver",
                                " F:\\ Selenium\\chromedriver-win32\\chromedriver.exe");
                WebDriver driver = new ChromeDriver();
                driver.manage().window().maximize();

                driver.get("http://webdriveruniversity.com/Dropdown-Checkboxes-
RadioButtons/index.html");
                driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
                // Get count of CheckBoxes present
                List<WebElement> checkboxes =
driver.findElements(By.xpath("//input[@type='checkbox']"));

                System.out.println(checkboxes.size() + " Number of CheckBoxes");
                // Get count of Dropdown menus
                List<WebElement> dropdown = driver.findElements(By.tagName("select"));

                System.out.println(dropdown.size() + " Number of DropDown Menus");
                // Get count no' of radio buttons
                List<WebElement> radioBtns =
driver.findElements(By.xpath("//input[@type='radio']"));
                System.out.println(radioBtns.size() + " Number of Radio Buttons");
                driver.close();
        }
}
```

Output:

4 Number of CheckBoxes

4 Number of DropDown Menus

8 Number of Radio Buttons

10. Black Box testing: (Functional Testing) Write and test a program to get the number of list items in a list / combo box

Drop.html

```
<html>
<title>Write and test a program to get the number of list items in a list / combo box.
</title>
<body>
<label >Select a Programming Language:</label>
<select name="language" id="language">
  <option value="javascript">JavaScript</option>
  <option value="python">Python</option>
  <option value="c++" disabled>C++</option>
  <option value="java" selected>Java</option>
</select>
</body>
</html>
```

```
package SeleniumWebDriver;
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 drop {
	public static void main(String[] args) {
		System.setProperty("webdriver.chrome.driver",
				" F:\\ Selenium\\chromedriver-win32\\chromedriver.exe ");
		WebDriver driver = new ChromeDriver();
		driver.manage().window().maximize();
		// URL launch
		driver.get("F:\\drop.html");
```

```
                // Identify list
                Select items = new Select(driver.findElement(By.name("language")));
                List<WebElement> mylist = items.getOptions();
                System.out.println("Number of items = " + mylist.size());
                driver.quit();
        }
}
```

OUTPUT:

Number of items = 4

11. Demonstrate how to handle Window Pop Up in Selenium WebDriver

```
package SeleniumWebDriver;
import org.openqa.selenium.Alert;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class windowPopUp {

        public static void main(String[] args) throws InterruptedException {
System.setProperty("webdriver.chrome.driver","F:\\ Selenium\\chromedriver-
win32\\chromedriver.exe");
                WebDriver driver = new ChromeDriver();

                driver.manage().window().maximize();

                driver.get("https://ksrtc.in/oprs-web/");
                driver.findElement(By.cssSelector("#toPlaceName")).click();

                Thread.sleep(2000);

                Alert simpleAlert = driver.switchTo().alert();
                String alertMessage= driver.switchTo().alert().getText();

                System.out.println(alertMessage);
                simpleAlert.accept();

                Thread.sleep(5000);
                driver.close();
        }
}
```

OUTPUT:

Please select Leaving From.

12. Write and test a program to count number of Check boxes on the page checked and unchecked

Checkbox.html

```
<html>
<body>
<h2>Choose your Hobbies</h2>
<input type="checkbox" name="hobby" value="Reading"> Reading<br>
<input type="checkbox" name="hobby" value="Swiming"> Swiming<br>
<input type="checkbox" name="hobby" value="Travelling"> Travelling<br>
<input type="checkbox" name="hobby" value="Hiking"> Hiking<br>
<input type="checkbox" name="hobby" value="Dancing"> Dancing<br>
<br><br>
</body>
</html>
```

Selenium script

```
package com.testing;

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 checkbox {

	public static void main(String[] args) throws InterruptedException {
		System.setProperty("webdriver.chrome.driver",
				"F:\\\\2023-24\\\\Odd\\\\Software Engineering and Testing\\\\SE and Testing Lab\\\\Selenium\\\\chromedriver-win32\\\\chromedriver.exe");
		WebDriver driver = new ChromeDriver();
		driver.get("F:\\2023-24\\Odd\\Software Engineering and Testing\\SE and Testing Lab\\Checkbox.html");

		driver.manage().window().maximize();
		Thread.sleep(3000);
		driver.findElement(By.xpath("/html/body/input[1]")).click();
		driver.findElement(By.xpath("/html/body/input[3]")).click();
```

```
                List <WebElement> checkBox =
driver.findElements(By.xpath("//input[@type='checkbox']"));
                int checkedCount = 0;
                int uncheckedCount = 0;
                for (int i=0;i<checkBox.size();i++)
                {
                 if ( checkBox.get(i).isSelected() == true)
                      checkedCount++;
                     else
                      uncheckedCount++;
                     }
                System.out.println("Number of checked checkboxes are " +checkedCount);
                System.out.println("Number of unchecked checkboxes are " +
uncheckedCount);


        driver.close();
        }


}
```

OUTPUT:

Number of checked checkboxes are 2

Number of unchecked checkboxes are 3

What Is TestNG?

TestNG, created by Cédric Beust, is one of Java's most popular test automation frameworks. The annotations, functionalities, usability, features, and ease of use provided by TestNG make it a prevalent choice of framework.

Benefits of using TestNG for Selenium automation testing:

- The reporting feature provides a detailed XML report of every successful, failed, or skipped test.
- Parallel testing assists testers in running multiple test cases.
- TestNG annotations feature allows developers to handle exceptions and understand the code easily.
- Test cases can be grouped or prioritized easily.
- Flexible runtime configuration.

Installing TestNG from Eclipse Marketplace

Step 1: Open Eclipse IDE. Once it is launched, navigate to **Help > Eclipse Marketplace**.

Step 2: Search for TestNG in the search bar and click on install for the first result.

Step 3: Make sure that all the checkboxes for TestNG are checked, and then click on the Confirm button.

Step 4: Accept the license and click on **Finish**.

Note: For the TestNG plugin to work, Eclipse must be restarted. When you right-click on a project, you can find TestNG in the menu options if installed.

Create TestNG Project In Eclipse

First of all, you will need to launch Eclipse and then follow the steps below to create a new TestNG project.

Step 1: Navigate to File > New > Java Project.

Step 2: Give your project a name for example, '**TestNG Project**' and click on Next.

Step 3: On the next screen, you will see the Java settings for your new project. To make it a TestNG project just navigate to the **Libraries** tab and click on the **Add Library** button.

Step 4: Choose **TestNG** from the list of libraries and click **Next**.

Step 5: Choose **JUnit** from the list of libraries and click **Next**.

Step 6: Now, you will see TestNG and JUnit added to your project libraries. Click on **Finish** and you are all set with your testNG project

Creating a TestNG class in Eclipse

Creating a TestNG class is as easy as creating a Java class.

Step 1: Navigate to src from the project folder and right-click the same. You will see TestNG as an option in the dropdown towards the bottom. Click on it and you will now see two sub-options to either create a TestNG class or convert the class to TestNG. As we are creating a new TestNG class, you need to select the first option.

Step 2: Generally, the source folder name is auto-filled but if it is not, you can simply browse through the same. Next, you can give any name to your class, for example,

'**TestNGTestOne**' and its package. For now, we will keep the basic annotations selected **@BeforeMethod** and **@AfterMethod**. However, Annotations can be configured at a later stage as well depending upon your test scenario

Step 3: You will now see a class(**TestNGTestOne.java**) in your project directory with default methods, viz f(), as well as beforeMethod() and afterMethod()

What is TestNG Annotation?

TestNG Annotation is a piece of code which is inserted inside a program or business logic used to control the flow of execution of test methods.

Benefits of using TestNG Annotations:

- TestNG Annotations made the life of testers very easy. Based on your requirements, you can access the test methods, i.e., it has no predefined pattern or format.
- You can pass the additional parameters to TestNG annotations.
- In the case of TestNG annotations, you do not need to extend any test classes.
- TestNG Annotations are strongly typed, i.e., errors are detected at the compile time.

Hierarchy of the TestNG Annotations:

- @BeforeSuite
- @BeforeTest
- @BeforeClass
- @BeforeMethod
- @Test
- @AfterMethod
- @AfterClass
- @AfterTest
- @AfterSuite

TestNG Annotation	Description
@BeforeSuite	The @BeforeSuite annotated method will run before the execution of all the test methods in the suite.
@AfterSuite	The @AfterSuite annotated method will run after the execution of all the test methods in the suite.
@BeforeTest	The @BeforeTest annotated method will be executed before the execution of all the test methods of available classes belonging to that folder.
@AfterTest	The @AfterTest annotated method will be executed after the execution of all the test methods of available classes belonging to that folder.
@BeforeClass	The @BeforeClass annotated method will be executed before the first method of the current class is invoked.
@AfterClass	The @AfterClass annotated method will be invoked after the execution of all the test methods of the current class.
@BeforeMethod	The @BeforeMethod annotated method will be executed before each test method will run.
@AfterMethod	The @AfterMethod annotated method will run after the execution of each test method.
@BeforeGroups	The @BeforeGroups annotated method run only once for a group before the execution of all test cases belonging to that group.
@AfterGroups	The @AfterGroups annotated method run only once for a group after the execution of all test cases belonging to that group.

What is Data Driven Framework

Data Driven framework is focused on separating the test scripts logic and the test data from each other. Allows us to create test automation scripts by passing different sets of test data. The test data set is kept in the external files or resources such as MS Excel Sheets, MS Access Tables, SQL Database, XML files etc., The test scripts connect to the external

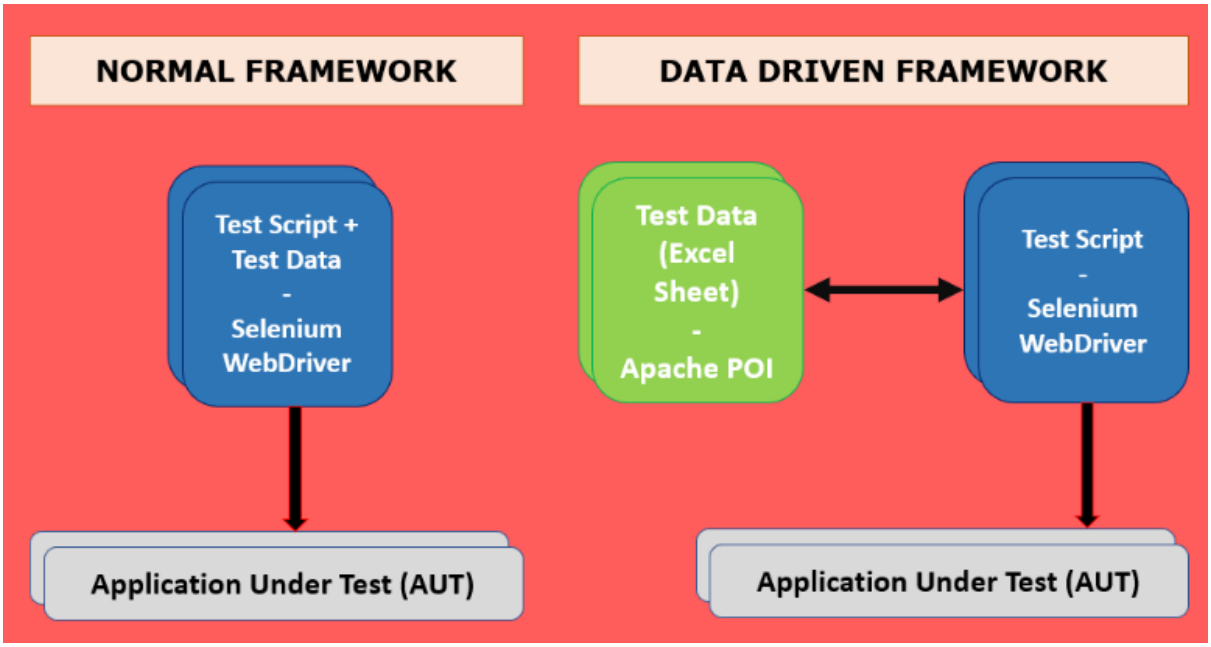
resources to get the test data. By using this framework we could easily make the test scripts work properly for different sets of test data. This framework significantly reduces the number of test scripts compared to a modular based framework.

Why Data Driven Framework

Usually, we place all our test data in excel sheets which we use in our test runs. Assume, we need to run a test script (Say, login test) with multiple test data. If we run the same test with multiple test data sets manually is time-consuming, and error-prone

Advantages of using Data Driven Test Framework

- Re-usability of code
- Improves test coverage
- Faster Execution
- Less maintenance
- Permits better error handling



13. **Write and test a program to update 10 student records into table into Excel file**

Prerequisite:

1) jxl jar file ,Downloaded from
https://sourceforge.net/projects/jexcelapi/files/jexcelapi/2.6.12/

2) An excel file with student records present in it. (In my case "Students.xls").

3) Install TestNG Plugin for eclipse ide

4) Open eclipse create a new java project - add a class - then add the jar file using Build path as shown in my previous video. Don't select any method here

5)Run the program as TestNG suite.
The status gets updated in Result.xls file as pass or fail.
Run the program and see the output on the specified location

```
package myProject;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import jxl.Sheet;
import jxl.Workbook;
import jxl.write.Label;
import jxl.write.WritableSheet;
import jxl.write.WritableWorkbook;
import org.testng.annotations.*;
public class studentRecord {
        @BeforeClass //@BeforeClass runs once before the entire test.
        public void setUp() throws Exception {}
        @Test
        public void testImportexport1() throws Exception {
        FileInputStream fi = new
FileInputStream("C:\\Users\\admin'\\Desktop\\StudentScore.xls");
        Workbook w = Workbook.getWorkbook(fi);
        Sheet s = w.getSheet(0);

        String a[][] = new String[s.getRows()][s.getColumns()];
        FileOutputStream fo = new
FileOutputStream("C:\\Users\\admin'\\Desktop\\Result.xls");

        WritableWorkbook wwb = Workbook.createWorkbook(fo);
        WritableSheet ws = wwb.createSheet("result1", 0);

        for (int i = 0; i < s.getRows(); i++)
        for (int j = 0; j < s.getColumns(); j++)
        {
        a[i][j] = s.getCell(j, i).getContents();
        Label l2 = new Label(j, i, a[i][j]);
        ws.addCell(l2);
        Label l1 = new Label(6, 0, "Result");
        ws.addCell(l1);
        }
        for (int i = 1; i < s.getRows(); i++) {
        for (int j = 2; j < s.getColumns(); j++)
        {
        a[i][j] = s.getCell(j, i).getContents();
        int x=Integer.parseInt(a[i][j]);
        if(x > 35)
        {
        Label l1 = new Label(6, i, "pass");
        ws.addCell(l1);
```

```
        }
        else
        {
        Label l1 = new Label(6, i, "fail");
        ws.addCell(l1);
        break;
        }
        }
        System.out.println("Records sucessfully updated ");
        }
        wwb.write();
        wwb.close();
        }
}
```

14. Write a program in java to test do…while construct using TestNG

SumOfNumbers.java

```
package dowhile;
public class SumOfNumbers {
        public static void main(String[] args) {
                int i = 1, sum = 0;
                do {
                sum += i; //sum=sum+i;
                i++;
                } while (i <= 10);
                System.out.println("The sum of numbers from 1 to 10 is: " + sum);
                }
                public static int calculateSum(int start, int end) {
                int i = start, sum = 0;
                do {
                sum += i;
                i++;
                } while (i <= end);
                                return sum;
                }
        }
```

SumOfNumbersTest.java

```
package dowhile.DoWhile;
import org.testng.Assert;
import org.testng.annotations.Test;
import dowhile.SumOfNumbers;
public class SumOfNumbersTest {
	@Test
	public void testPositiveValuesInRange() {
	int expectedSum = 55;
	int actualSum=SumOfNumbers.calculateSum(1,10);
	Assert.assertEquals(actualSum,expectedSum);
	}
	@Test
	public void testNegativeValuesInRange() {
	int expectedSum =0;
	int actualSum=SumOfNumbers.calculateSum(-10,10);
	Assert.assertEquals(actualSum,expectedSum);
	}
	@Test
	public void testOutOfRange() {
		//int expectedSum=155
	int expectedSum = 0;
	int actualSum =SumOfNumbers.calculateSum(11,20);
	Assert.assertEquals(actualSum,expectedSum);
	}
}
```

OUTPUT:

```
PASSED: dowhile.DoWhile.SumOfNumbersTest.testPositiveValuesInRange
PASSED: dowhile.DoWhile.SumOfNumbersTest.testNegativeValuesInRange
FAILED: dowhile.DoWhile.SumOfNumbersTest.testOutOfRange
===============================================
   Default test
```

```
Tests run: 3, Failures: 1, Skips: 0
===============================================


===============================================
Default suite
Total tests run: 3, Passes: 2, Failures: 1, Skips: 0
===============================================
```

15. Write a program in java to test for loop construct using TestNG

FibonacciSeries.java

```
package forLoop;
public class FibonacciSeries {
	public static void main(String[] args)
	{
		int limit = 50;
		int num1 = 0, num2 = 1;
		System.out.print(num1 + " " + num2 + " ");
		for(int i = 2; i <= limit; i++)
		{
			int sum = num1 + num2;
			System.out.print(sum + " ");
			num1 = num2;
			num2 = sum;
		}
	}		public static int[] generateFibonacciSeries(int n)
		{
			int[] series = new int[n];
			series[0] = 0;
			if (n > 1)
			{
				series[1] = 1;
```

```
                    for (int i = 2; i < n; i++)
                    {
                        series[i] = series[i - 1] + series[i - 2];
                    }
                }
            return series;
            }
}
```

FibonacciSeriesTest.java

```
package FibonacciSeriesTest;
import org.testng.Assert;
import org.testng.annotations.Test;
import forLoop.FibonacciSeries;
public class FibonacciSeriesTest {
    @Test
    public void testFibonacciSeriesWithLimit5() {
    int[] expectedOutput = {0, 1, 1, 2, 3};
    int limit = 5;
    int[] actualOutput = FibonacciSeries.generateFibonacciSeries(limit);
    Assert.assertEquals(actualOutput, expectedOutput);
    }
    @Test
    public void testFibonacciSeriesWithLimit10()
    {
    int[] expectedOutput = {0, 1, 1, 2, 3, 5, 8,13, 21, 34};
    int limit = 10;
    int[] actualOutput = FibonacciSeries.generateFibonacciSeries(limit);
    Assert.assertEquals(actualOutput,expectedOutput);
    }
    @Test
    public void testFibonacciSeriesWithLimit1() {
```

```
        int[] expectedOutput = {0};
        int limit = 1;
        int[] actualOutput = FibonacciSeries.generateFibonacciSeries(limit);
        Assert.assertEquals(actualOutput,
        expectedOutput);
        }
        @Test
        public void
        testFibonacciSeriesWithNegativeLimit() {
        int[] expectedOutput = {};
        int limit = -5;
        int[] actualOutput =FibonacciSeries.generateFibonacciSeries(limit);
        Assert.assertEquals(actualOutput,
        expectedOutput);
        }
}
```

OUTPUT:

```
PASSED: FibonacciSeriesTest.FibonacciSeriesTest.testFibonacciSeriesWithLimit10
PASSED: FibonacciSeriesTest.FibonacciSeriesTest.testFibonacciSeriesWithLimit5
PASSED: FibonacciSeriesTest.FibonacciSeriesTest.testFibonacciSeriesWithLimit1
FAILED: FibonacciSeriesTest.FibonacciSeriesTest.testFibonacciSeriesWithNegativeLimit
===============================================
   Default test
   Tests run: 4, Failures: 1, Skips: 0
===============================================
===============================================
Default suite
Total tests run: 4, Passes: 3, Failures: 1, Skips: 0
```