# STQA Practical

## INTRODUCTION TO SELENIUM

### 1. History of Selenium

• In 2004 invented by Jason R. Huggins and team.
• Original name is JavaScript Functional Tester [JSFT]
• Open source browser based integration test framework built originally by Thought works.
• 100% JavaScript and HTML
• Web testing tool
• That supports testing Web 2.0 applications
• Supports for Cross-Browser Testing (ON Multiple Browsers)
• And multiple Operating Systems
• Cross browser – IE 6/7, Firefox .8+, Opera, and Safari 2.0+

### 2. What is Selenium?

• Acceptance testing tool for web-apps
• Tests run directly in browser
• Selenium can be deployed on Windows, Linux, and Macintosh.
• Implemented entirely using browser technologies -
 JavaScript
 DHTML
 Frames

### 3. Selenium Components

– Selenium IDE
**Created by Shinya Kasatani** of Japan
– Selenium RC
**Created by** Paul Hammant
– Selenium Grid
**Developed by** Patrick Lightbody

### 3.1 Selenium IDE

• The Selenium-IDE (Integrated Development Environment) is the tool you use to develop your Selenium test cases.
• It is Firefox plug-in
• Firefox extension which allows record/play testing paradigm
• Automates commands, but asserts must be entered by hand

• Creates the simplest possible Locator
• Based on Selenese

---

**Practical 1**

1. **Selenium IDE Demo Installation**
   **Version to be used**
   (Selenium IDE Version 3.1.1 works with Firefox 56.0 and later)

   **You need**
   - Mozilla Firefox
   - Active Internet Connection
   - If you do not have Mozilla Firefox yet, you can download it **from http://www.mozilla.org/en-US/firefox/new.**

**STEPS:**

1. Launch **Mozilla Firefox** Browser.
2. Type URL https://www.seleniumhq.org/download/
   **OR**
   https://addons.mozilla.org/en-US/firefox/addon/selenium-ide/
3. Selenium IDE Add-ons page will get open then Click on Add to Firefox button
4. Firefox will show one popup saying do you want to allow Mozilla Firefox to install Selenium IDE Add-ons or not. **Click** on **Install** button.
5. Firefox will automatically install Selenium IDE software. After the installation is completed, a pop up window appears asking to re-start the Firefox. Click on the "Restart Now" button to reflect the Selenium IDE installation. Click on Restart Now button.
6. On clicking on the Restart Now button, Firefox will restart automatically. In case you missed the pop-up, simply close the Firefox and launch again.
7. Once the Firefox is booted and started again, we can see selenium IDE under the tools menu list. **Selenium IDE icon** will be displayed in the Firefox toolbar.
8. Click on **Selenium IDE icon to open Selenium IDE.**
   - Selenium IDE Works with all major versions, but we recommend to use 47.0.1 & above as they have better stability.
   - Selenium IDE is no longer compatible with Firefox 55 and above.

**4 test cases for 4 websites**

i. http://store.demoqa.com
ii. https://www.seleniumhq.org
iii. htttp://www.google.com
iv. htttp://www.yahoo.com
v. http://demo.guru99.com/test/newtours/

**Step 1**

- Launch Firefox and Selenium IDE.
- Type the value for our Base URL: http://demo.guru99.com/test/newtours/.
- Toggle the Record button on (if it is not yet toggled on by default).

**Step 2**

- In Firefox, navigate to http://demo.guru99.com/test/newtours/. Firefox should take you to the page similar to the one shown below.

**Step 3**

- Right-click on any blank space within the page, like on the Mercury Tours logo on the upper left corner. This will bring up the Selenium IDE context menu. Note: Do not click on any hyperlinked objects or images
- Select the "Show Available Commands" option.
- Then, select "assertTitle exact: Welcome: Mercury Tours." This is a command that makes sure that the page title is correct.

**Step 4**

- In the "User Name" text box of Mercury Tours, type an invalid username, "invalidUNN".
- In the "Password" text box, type an invalid password, "invalidPWD".

**Step 5**

- Click on the "Sign-In" button. Firefox should take you to this page.

**Step 6**

- Toggle the record button off to stop recording.

**Step 7**

- Now that we are done with our test script, we shall save it in a test case.

**Step 8**

- Choose your desired location, and then name the Test Case as "Invalid_login".
- Click the "Save" button.

**Step 9**

- Notice that the file was saved as side extension.

**Step 10**

- Go back to Selenium IDE and click the Playback button to execute the whole script. Selenium IDE should be able to replicate everything properly.

**Practical 2**

2. **Conduct a test suite for any two web sites.**

**Practical 3**

3. **Install Selenium server and demonstrate it using a script in Java/PHP.**

**Introduction**

1. **Selenium-RC** is the solution for tests that need more than simple browser actions and linear execution.
2. Selenium-RC uses the full power of programming languages to create more complex tests like reading and writing files, querying a database, emailing test results.
3. You'll want to use Selenium-RC whenever your test requires logic not supported by Selenium-IDE.
4. What logic could this be? For example, Selenium-IDE does not directly support:

- Condition statements
- Iteration
- Logging and reporting of test results
- Error handling, particularly unexpected errors
- Database testing
- Test case grouping
- Re-execution of failed tests
- Test case dependency
- Screenshot capture of test failures

5. Although these tasks are not supported by Selenium directly, all of them can be achieved by using programming techniques with a language-specific Selenium-RC client library.

**Installation of Selenium RC and Eclipse**

**Download Eclipse**

1. Go to URL – http://www.eclipse.org/downloads/
2. Select **Eclipse IDE for Java Developers** (Click on Windows 32 bit platform)
3. Click on OK button and save to a local drive (i.e. C: or D:, etc)
4. Unzip the downloaded zip file and rename that to Eclipse
5. Create one more folder "Eclipse-Workspace" (i.e. C:Eclipse-Workspace)in the same drive where Eclipse is unzipped and renamed.

6. Create Eclipse desktop shortcut (go to C:Eclipse folder –> right click Eclipse.exe and then click on “desktop create shortcut”).

1. Now we need to create a workspace folder –> C:Eclipse-WorkspaceSeleniumTests
2. Double click on “Eclipse shortcut on Desktop”
3. This opens the Eclipse
4. Close Eclipse welcome screen
5. Click File menu –> Launch Worspace –> other
6. Now Select the C:Eclipse-WorkspaceSeleniumTests folder.

We have finished setting up the eclipse.

Now, we need to download Selenium RC server / client driver and configure that to Eclipse
1. Download Selenium server: http://seleniumhq.org/download/
2. Download Selenium Client driver for Java (from Selenium Client Drivers section)
3. Create “Selenium” folder in C: drive and copy the Selenium-server.jar as well as unzip the Selenium Client driver (C:Selenium)

Downloading and unzipping the files into a folder is done.
We need to configure the appropriate Selenium Client driver Jar file to the Eclipse.

1. Go to Eclipse –> Click File –> New –> Project (from various options need to select just “project”)
2. In Select Wizard –> Click Java –> “Java Project”
3. Give the project name (e.g. DemoTests)
4. Click Finish – Click Yes
5. Now we are done with creation of project and need to configure the Selenium Client driver to this Project
6. Right Click “DemoTests” project
7. Click “Java Build Path”
8. Click Libraries tab
9. Click “Add External JARs” button
10. Select “Selenium Client Drivers” unzipped in C:Selenium folder (Selenium Server JAR file should not be added)
11. Click OK
12. Referenced libraries –> contains both the Selenium Client driver jar files.

13. Create a new class file as “SeleniumDemo” in the “DemoTest” by right click on src folder.
14. Copy the below code in the class file:-

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.remote.DesiredCapabilities;

public class hello
{
	static String driverPath = "D:\\selenium pracs\\geckodriver-v0.21.0-win32\\GeckoDriver.exe";
   public static WebDriver driver;
	public static void main(String args[])
	{
		int a=10,b=20;
		System.out.println("Hi....");
		System.out.println(a+b);
		System.out.println("Selenium demo.....");

		System.setProperty("webdriver.gecko.driver",driverPath);
		DesiredCapabilities capabilities = DesiredCapabilities.firefox();
		capabilities.setCapability("marionette",true);
		driver= new FirefoxDriver(capabilities);
		driver.get("https://www.facebook.com/");
		driver.manage().window().maximize();
		driver.quit();

	}
}
```

**(If Firefox is unable to open, download geckodriver from the website, https://github.com/mozilla/geckodriver/releases according to the operating system, unzip the folder and keep the file along with gecko folder in the same folder wherever selenium jar files folder is there)**

4. **Write and test a program to login a specific web page. (Using JUnit)**

**PART 1) Download JUnit using**

Step 1) **https://github.com/junit-team/junit4/wiki/Download-and-Install**

**OR**

**Visit http://junit.org/junit4/ and click Download and Install**

**(Downloading junit jar files and put in the junit folder along folder wherever selenium jar files folder is there)**

**JUnit Environment Setup**

**Step 1)** You need to set **JUNIT_HOME** environment variable to point out the base location where you have placed JUnit Jars.

For example, if you have created a JUnit folder in c: drive and placed jars there, then for environment settings you need to open control panel ->advanced ->environment variable.

1. Under environment window clicks on "new" button.
2. When you click on new button in environment variables, it will open another window
3. **Step 2)** A "New System Variable" window will open:
4. Provide variable name as "JUNIT_HOME".
5. Provide JUnit value as JUnit path where you have copied JUnit jar files.
6. Click on OK.

When you click on OK, it will create a new system variable with the given name and value.

**Step 3)** After creating JUNIT_HOME, create another variable with the name CLASSPATH. Again go to Environment Variables and follow the below steps.

1. Click on "new" button. When you click on new in environment variables, it will open another window.

**Step 4)** In this step, point out JUNIT_HOME to JUnit.jar which is placed in JUnit folder as given below:

1. Variable Name: JUNIT_HOME
2. Variable Value: %CLASSPATH%;%JUNIT_HOME%\JUnit4.10.jar;.;

3. Click on the OK button.

Step 5) **Once you click on the 'OK' button, you can verify that a new environment variable named "CLASSPATH" can be seen under system variable.**

**PART 2) Install JUnit jar file in eclipse**
**Step 1)** Right click on project:

1. Click on "build path" and then
2. Click on "Configure build path".

**Step 2)** In this step,

1. Go to java build path window as shown in below figure
2. Now click on "Add External JARs" button to add your downloaded JUnit.jar file with eclipse.

After adding a JUnit.jar file, click on 'OK' button to close java build path window.

**Part 3) Verifying whether required jar file for JUnit is in my build path**
In order to verify JUnit jar file in eclipse, you need to follow below-mentioned steps:

1. Right click on project -> Build Path
2. Click on "Configure build path".
3. **Step 2)** Java build path window will appear as shown below.
4. In that window, go to Libraries tab to see all jar files. In jar file tree view, you need to look for the jar file name which is starting with JUnit.
5. Once you expand JUnit libraries, you can see java libraries
6. Now you are ready to use JUnit with eclipse.

**PART 4) Verify JUnit setup**
7. You can create a simple JUnit test to verify JUnit setup. See below test class:
8. **Step 1)** Create a java class named TestJUnit.java and provide a simple assert statement.

```
import org.junit.Test;
import static org.junit.Assert.assertEquals;
public class TestJunit {
  @Test
```

```
  public void testSetup() {
    String str= "I am done with Junit setup";
    assertEquals("I am done with Junit setup",str);
  }
}
```

**Step 2)** Create a Test Runner class to execute above test.

```
import org.junit.runner.JUnitCore;
import org.junit.runner.Result;
import org.junit.runner.notification.Failure;

public class TestRunner {
  public static void main(String[] args) {
    Result result = JUnitCore.runClasses(TestJunit.class);
    for (Failure failure : result.getFailures()) {
      System.out.println(failure.toString());
    }
    System.out.println("Result=="+result.wasSuccessful());
  }
}
```

**Step 3)** to execute the test, follow below steps:

1. Right click on TestRunner.java and click on "Run As" as shown below
2. Another window will be open once you click on "Run As", click on "1 JUnit Test"

**Step 4)** here is the output or result for your test. If it is successfully executed, it will show a green check mark in front of it.

**Wordpresscode**

```
package package1;
import java.util.concurrent.TimeUnit;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.testng.Assert;
import org.testng.annotations.AfterMethod;
import org.testng.annotations.DataProvider;
import org.testng.annotations.Test;
import org.openqa.selenium.firefox.FirefoxDriver;
```

```
//import DDT.FirefoxDriver;
public class WordPressNew {
	FirefoxDriver driver; //global variable
	@Test(dataProvider="wordpressData")
	public void loginToWordpress(String username,String password) throws
InterruptedException
	{
	System.setProperty("webdriver.gecko.driver",
"E:\\seleniumsettings\\gecko\\geckodriver.exe");
	//FirefoxDriver driver = new FirefoxDriver(); as its local , to make it global has
been mentioned above
	driver = new FirefoxDriver();
	driver.manage().window().maximize();
	driver.manage().timeouts().implicitlyWait(20,TimeUnit.SECONDS);
	driver.get("http://demosite.center/wordpress/wp-login.php");
	driver.findElement(By.id("user_login")).sendKeys("username");
	driver.findElement(By.id("user_pass")).sendKeys("password");
	driver.findElement(By.xpath("//*[@id=\"wp-submit\"]")).click();
	Thread.sleep(5000);
	//System.out.println(driver.getTitle());
	//till here it will show 3 runs and 0 failures for admin1, admin2 and admin3
until we dont provide any validations,
	//the task would depend on script
	//for this we will use assert to show validations
	Assert.assertTrue(driver.getTitle().contains("Dashboard"),"User is not able to
login:invalid Credential");
	//if above assertion failes then only User is not able to login would be printed
	System.out.println("Page title is verified:User is able to login successfullly");
	}
	@AfterMethod
	public void tearDown() //if testcase fails then teardown() would get exec
	{
		driver.quit();
	}
	@DataProvider(name="wordpressData")
```

```
public Object[][] passData() // give atleast one  username/password which is a
running...
{
        Object[][] data=new Object[3][2];
        data[0][0]="admin1";

        data[0][1]="demo1";
        data[1][0]="admin";
        data[1][1]="demo123";
        data[2][0]="admin2";
        data[0][1]="admin1234";
        return data;
}
}
```

5. **Write and test a program to update 10 student records into table into Excel file (using TestNG)**

TestNG is a testing framework inspired from JUnit and NUnit but introducing some new functionality that make it more powerful and easier to use, such as:

- Annotations.
- Run your tests in arbitrarily big thread pools with various policies available (all methods in their own thread, one thread per test class, etc...).
- Test that your code is multithread safe.
- Flexible test configuration.
- Support for data-driven testing (with @DataProvider).
- Support for parameters.
- Powerful execution model (no more TestSuite).
- Supported by a variety of tools and plug-ins (Eclipse, IDEA, Maven, etc...).
- Embeds BeanShell for further flexibility.
- Default JDK functions for runtime and logging (no dependencies).
- Dependent methods for application server testing.

**TestNG is designed to cover all categories of tests: unit, functional, end-to-end, integration, etc...**

## Installing TestNG in eclipse

1. Select Help / Software updates / Find and Install.
2. Search for new features to install.
3. New remote site.
4. For Eclipse 3.4 and above, enter http://beust.com/eclipse.
5. For Eclipse 3.3 and below, enter http://beust.com/eclipse1.
6. Make sure the check box next to URL is checked and click Next.
7. **Eclipse will then guide you through the process.**

## Launching your tests in Eclipse

• We finished writing our tests, now how can we run them?

• You can launch TestNG from the command line, using an Eclipse plugin or even programatically. We are going to use the Eclipse plugin. Follow the steps described on the official TestNG documentation over here.

• If you installed TestNG correctly, you will see this menu when you right click on the XML file:


Click on "Run as TestNG Suite" and your test will start running.

## Selenium Tests with Microsoft Excel

Parameterizing a test from external sources such as Microsoft Excel is always recommended in order to handle large amount of test data. To read data from Excel, we need APIs which support opening file, reading data, and writing data into Excel. We should know various classes and methods which support above mentioned operations. In this post, let us try to figure out which is the API that supports all the activities we need to do during execution of a test.

Jxl.jar is an open source Java API which supports read Excel spreadsheets and to write into Excel spreadsheets. Below are some of the operations that we can handle with this API.

1. Read data from Excel spreadsheet
2. Read and write formulas into spreadsheets
3. Generate spreadsheets
4. Supports formatting of font, number, and date
5. Supports coloring of cells

To access the methods and classes provided by this API inside Eclipse we need to add this JAR file to the Java Build Path.
**(I have explained steps to add external Jar files to Java Build Path in previous examples)**

**Download the jxl.jar from "http://jexcelapi.sourceforge.net/"**
Add the JAR file to Java Build Path
Add import statements to the .java file as below to read from an Excel spreadsheet

```
import jxl.Cell;
import jxl.Sheet;
import jxl.Workbook;
import jxl.read.biff.BiffException;
```

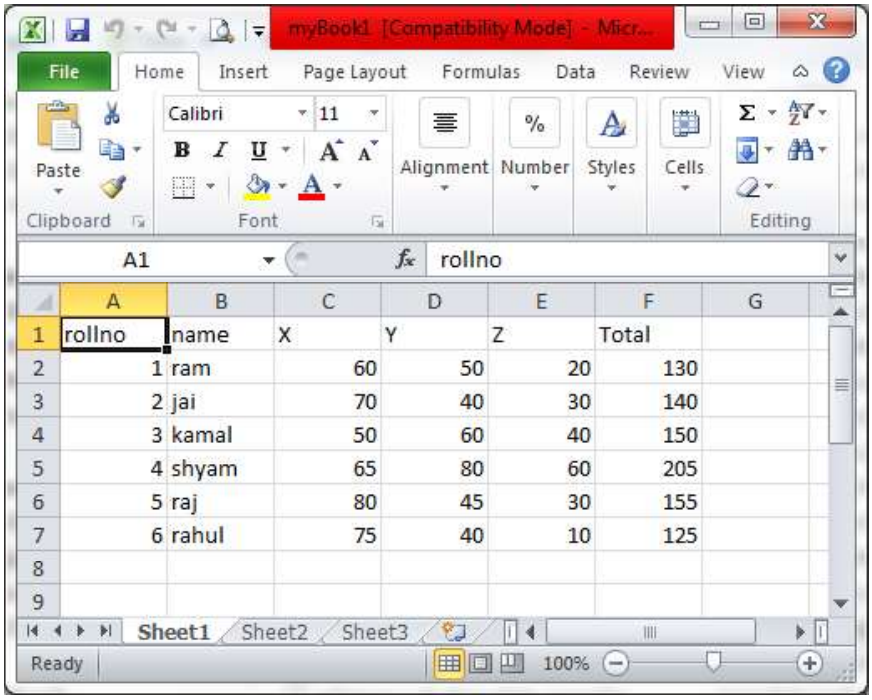
**//Code to update 10 student records into table into Excel file**

```
import org.testng.annotations.BeforeClass;
import org.testng.annotations.Test;
import jxl.*;
import jxl.read.*;
import jxl.write.*;
import java.io.*;

public class updatestudrecords {
	@BeforeClass
	public void f1()
	{}
	@Test
	public void testImportexport1() throws Exception {
	FileInputStream fi = new FileInputStream("D:\\selenium pracs\\myBook1.xls");
	Workbook w = Workbook.getWorkbook(fi);
	Sheet s = w.getSheet(0);
	String a[][] = new String[s.getRows()][s.getColumns()];
	FileOutputStream fo = new FileOutputStream("D:\\selenium
pracs\\myBook1res.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++)
```

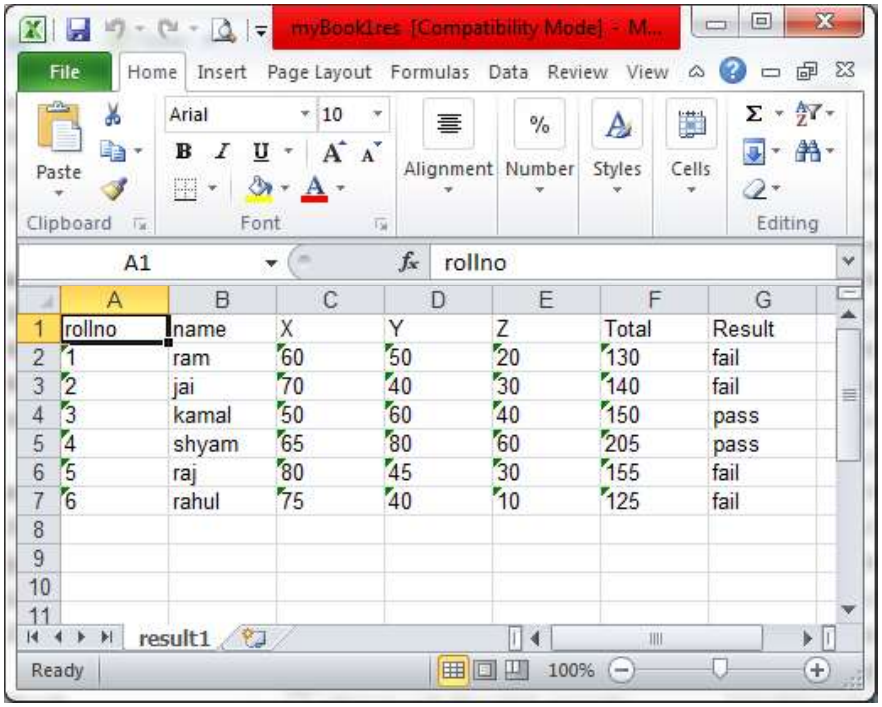
```
{
a[i][j] = s.getCell(j, i).getContents();
Label l2 = new Label(j, i, a[i][j]);
ws.addCell(l2);
Label l1 = new Label(6, 0, "Result");
ws.addCell(l1);
}
for (int i = 1; i < s.getRows(); i++) {
        for (int j = 2; j < s.getColumns(); j++)
        {
        a[i][j] = s.getCell(j, i).getContents();
        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;   }              }
        }
        wwb.write();
        wwb.close();          }       }
```

**Input:-**




| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | rollno | name | X | Y | Z | Total | |
| 2 | 1 | ram | 60 | 50 | 20 | 130 | |
| 3 | 2 | jai | 70 | 40 | 30 | 140 | |
| 4 | 3 | kamal | 50 | 60 | 40 | 150 | |
| 5 | 4 | shyam | 65 | 80 | 60 | 205 | |
| 6 | 5 | raj | 80 | 45 | 30 | 155 | |
| 7 | 6 | rahul | 75 | 40 | 10 | 125 | |
| 8 | | | | | | | |
| 9 | | | | | | | |

Output:-

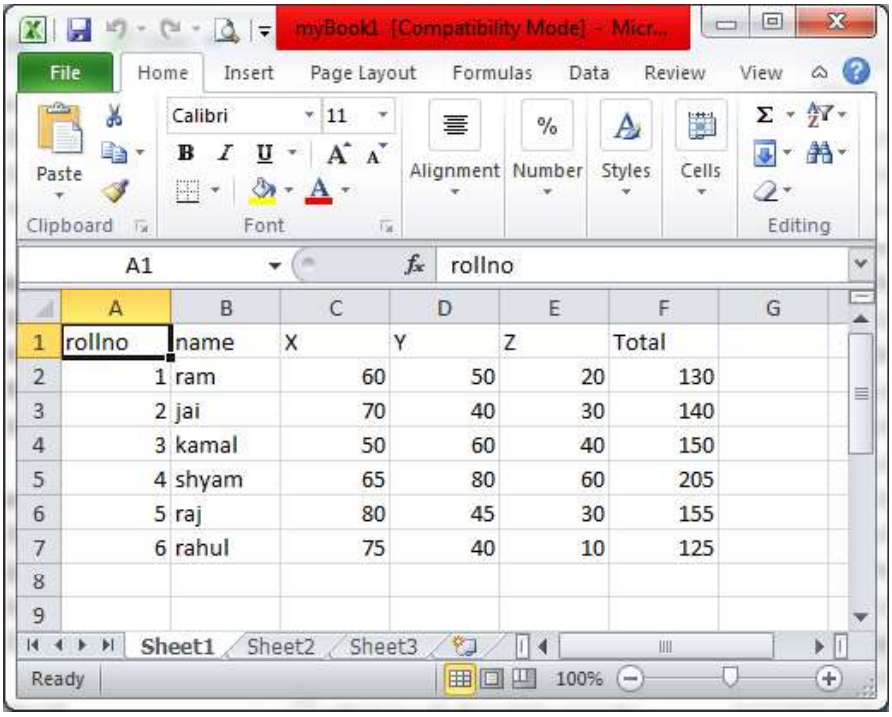



| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | rollno | name | X | Y | Z | Total | Result |
| 2 | 1 | ram | 60 | 50 | 20 | 130 | fail |
| 3 | 2 | jai | 70 | 40 | 30 | 140 | fail |
| 4 | 3 | kamal | 50 | 60 | 40 | 150 | pass |
| 5 | 4 | shyam | 65 | 80 | 60 | 205 | pass |
| 6 | 5 | raj | 80 | 45 | 30 | 155 | fail |
| 7 | 6 | rahul | 75 | 40 | 10 | 125 | fail |
| 8 | | | | | | | |
| 9 | | | | | | | |
| 10 | | | | | | | |
| 11 | | | | | | | |

6. **Write and test a program to select the number of students who have scored more than 60 in any one subject (or all subjects).**

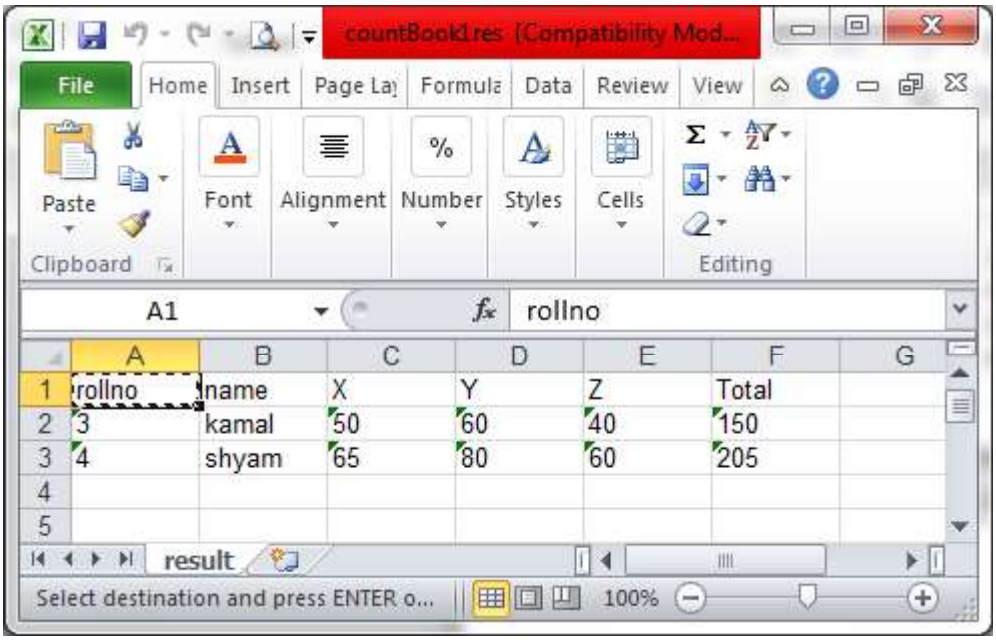
```
import jxl.*;
import jxl.read.*;
import jxl.write.*;
import java.io.*;
import org.testng.annotations.Test;
public class countstuds {
	@Test
	public void testImportexport1() throws Exception {
	FileInputStream fi = new FileInputStream("D:\\selenium pracs\\myBook1.xls");
	Workbook w = Workbook.getWorkbook(fi);
	Sheet s = w.getSheet(0);
	String a[][] = new String[s.getRows()][s.getColumns()];
	FileOutputStream fo = new FileOutputStream("D:\\selenium
pracs\\countBook1res.xls");
	WritableWorkbook wwb = Workbook.createWorkbook(fo);
	WritableSheet ws = wwb.createSheet("result", 0);
	int c=0;
	for (int i = 0; i < s.getRows(); i++) {
	for (int j = 0; j < s.getColumns(); j++)
	{
	if(i >= 1)
	{	String b= new String();
	b=s.getCell(3,i).getContents();
	int x= Integer.parseInt(b);
	if( x < 60)
	{	c++;
	break;		}
	}
	a[i][j] = s.getCell(j, i).getContents();
	Label l2 = new Label(j, i-c, a[i][j]);
	ws.addCell(l2);
	}  }
	wwb.write();
	wwb.close();
	}   }
```

**Input:-**




| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | rollno | name | X | Y | Z | Total | |
| 2 | 1 | ram | 60 | 50 | 20 | 130 | |
| 3 | 2 | jai | 70 | 40 | 30 | 140 | |
| 4 | 3 | kamal | 50 | 60 | 40 | 150 | |
| 5 | 4 | shyam | 65 | 80 | 60 | 205 | |
| 6 | 5 | raj | 80 | 45 | 30 | 155 | |
| 7 | 6 | rahul | 75 | 40 | 10 | 125 | |
| 8 | | | | | | | |
| 9 | | | | | | | |

Output:-




| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | rollno | name | X | Y | Z | Total | |
| 2 | 3 | kamal | 50 | 60 | 40 | 150 | |
| 3 | 4 | shyam | 65 | 80 | 60 | 205 | |
| 4 | | | | | | | |
| 5 | | | | | | | |

7. **Write and test a program to provide total number of objects present / available on the page.**

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.remote.DesiredCapabilities;

public class nlinks
{
        static String driverPath = "D:\\selenium pracs\\geckodriver-v0.21.0-
win32\\GeckoDriver.exe";
   public static WebDriver driver;
        public static void main(String args[])
        {

           System.setProperty("webdriver.gecko.driver",driverPath);
                DesiredCapabilities capabilities = DesiredCapabilities.firefox();
                capabilities.setCapability("marionette",true);
                driver= new FirefoxDriver(capabilities);
                driver.get("http://toolsqa.wpengine.com/");


        java.util.List<WebElement> links =
driver.findElements(By.tagName("a"));
         System.out.println("Total links are"+links.size());
        for (int i = 0; i<links.size(); i=i+1)
        {
        System.out.println("Link  "+ i + "    Link name   "+ links.get(i).getText());
        }
}
}
```

**Output:-**

Total links are 247

8. **Write and test a program to get the number of items in a list / combo box.**

9. **Write and test a program to count the number of check boxes on the page checked and unchecked count.**

10. **Load Testing using JMeter, Android Application testing using Appium Tools, Bugzilla Bug tracking tools.**

    **http://jmeter.apache.org/download_jmeter.cgi#binaries**
    **https://jmeter.apache.org/download_jmeter.cgi**
    **https://sourceforge.net/projects/jexcelapi/files/jexcelapi/2.6.12/jexcelapi_2_6_12.zip/download**

    **https://www.softwaretestinghelp.com/appium-tutorial-how-to-automate-android-apps-on-an-ios-system/**

    **http://toolsqa.com/mobile-automation/appium/install-android-sdk-adb-on-windows/**

    **https://www.bugzilla.org/**

    **https://www.bugzilla.org/news/#release504**