Name :- Rohan Babulnath Kamble

Class :- T. Y. B. Sc. CS

PRN NO :- 2020420004

Subject :- Software Testing And Quality Assurance (USCSP503)

INDEX

Practical No :- 1

Aim:- Install Selenium IDE; Write a test suite containing minimum 4 test cases for different formats.

Steps:-

How to Download & Install Selenium IDE for Firefox Installation of Selenium IDE

What you need

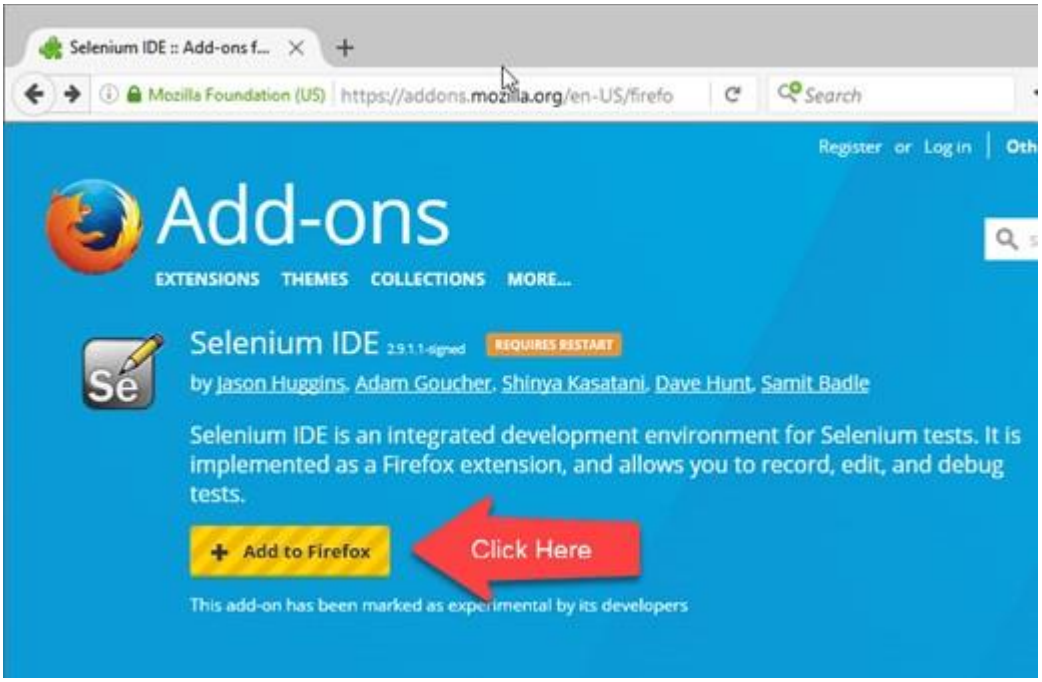
- Mozilla Firefox
- Active Internet Connection

If you do not have Mozilla Firefox yet, you can download it from http://www.mozilla.org/enUS/firefox/new.

Selenium IDE Works with all major versions, but we recommend to use 47.0.1 & above as they have better

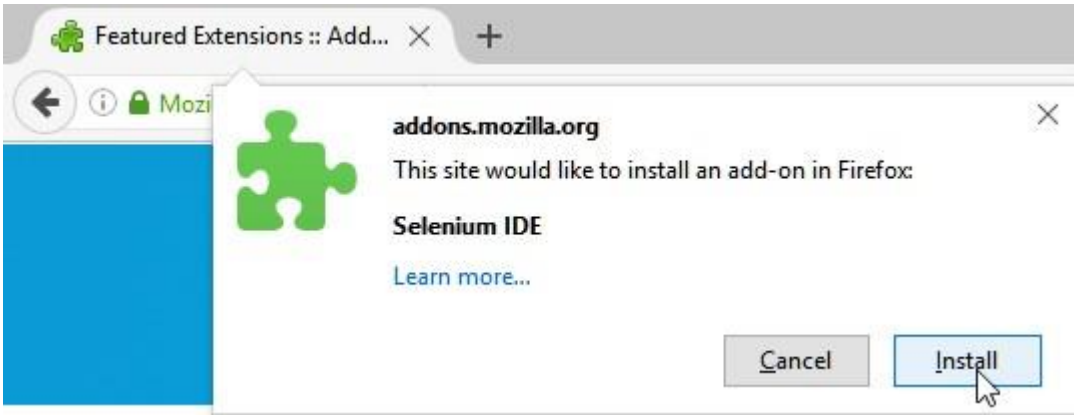
Step - 1)

Launch Firefox and navigate to https://addons.mozilla.org/en US/firefox/addon/selenium-ide/. Click on Add to Firefox



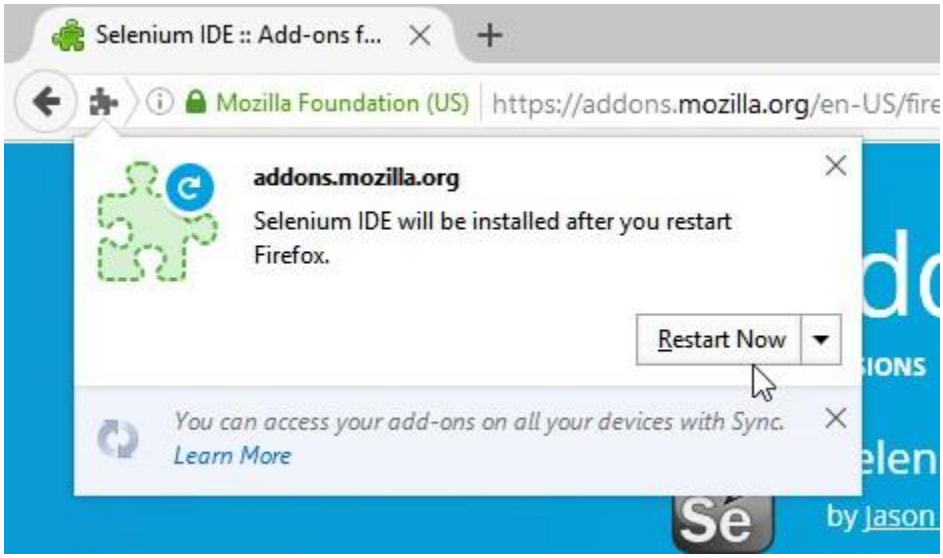
Steps - 2)

Wait until Firefox completes the download and then click "**Install.**"



Step - 3)

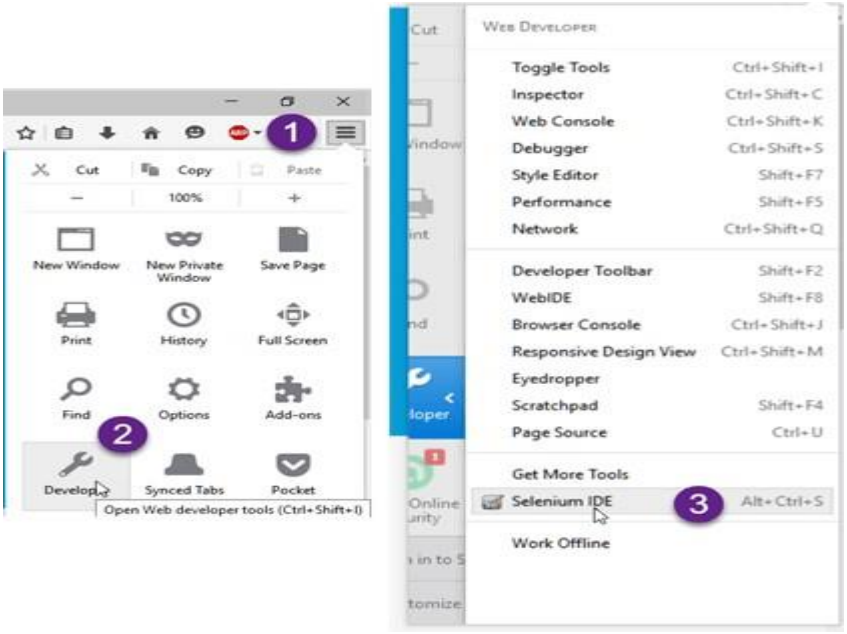
Wait until the installation is completed. In the pop-up window, click "**Restart Now**."



Step - 4)

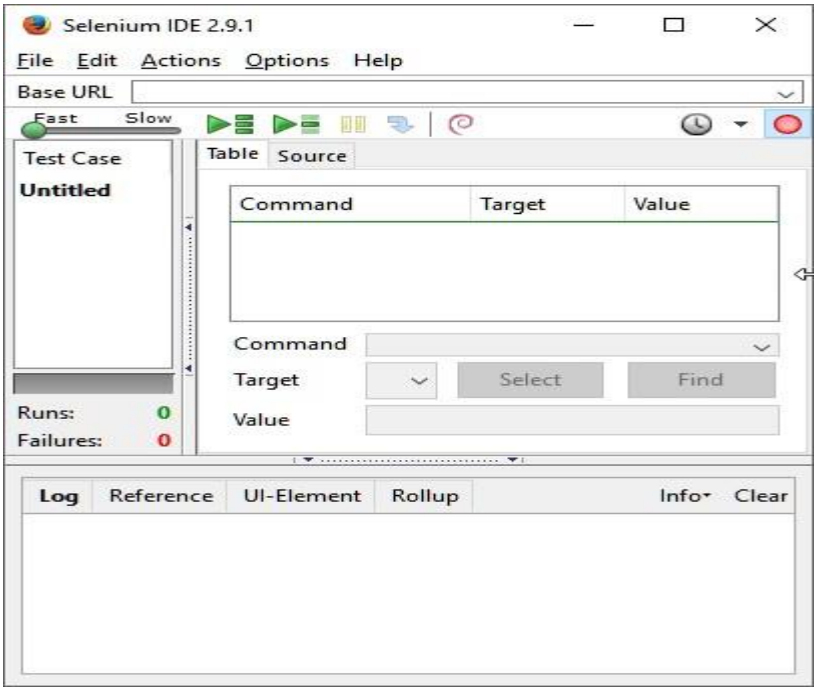
After Firefox has restarted, **launch Selenium IDE** using either of two ways:

- **By pressing Ctrl+Alt+S**
- **By clicking on the Firefox menu button> Developer>Selenium IDE**



Step - 5)

Selenium IDE should launch as shown below

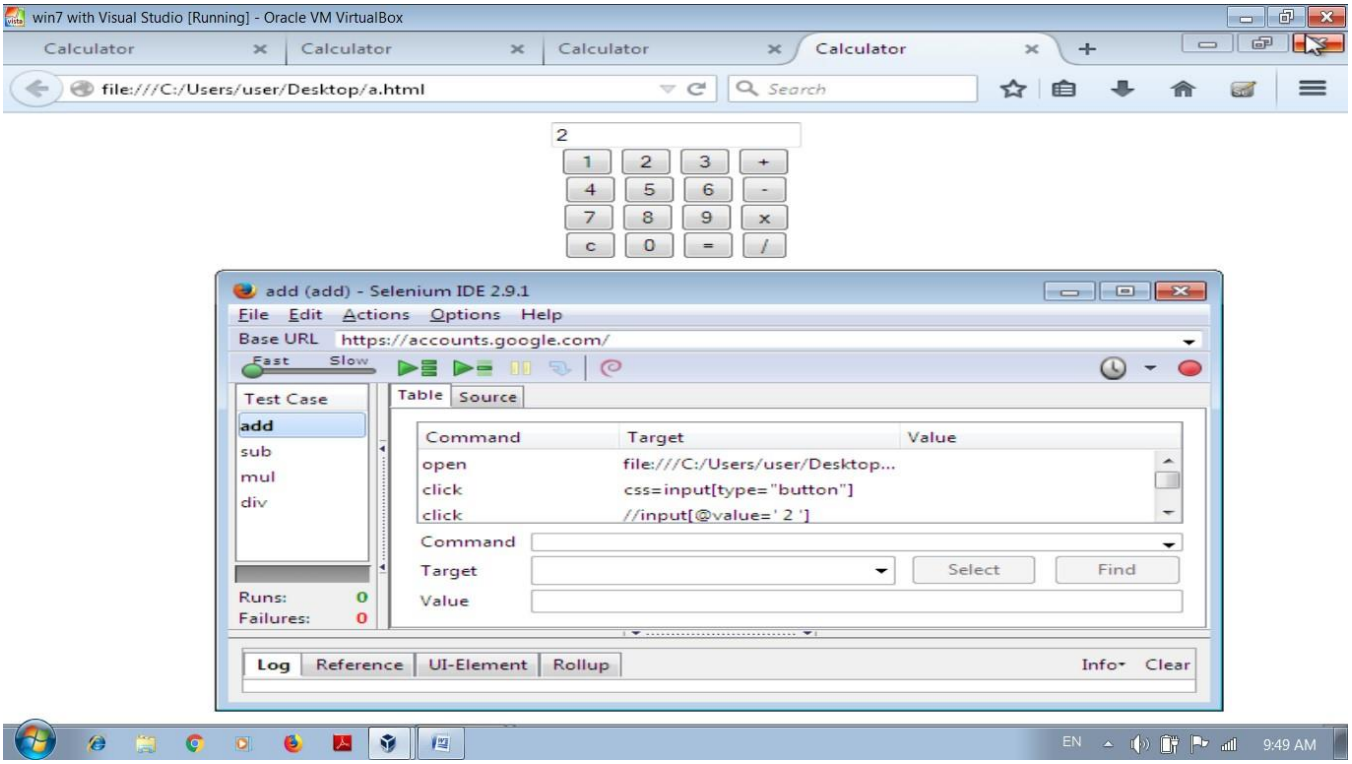


Write a test suite containing minimum 4 test cases for different formats.

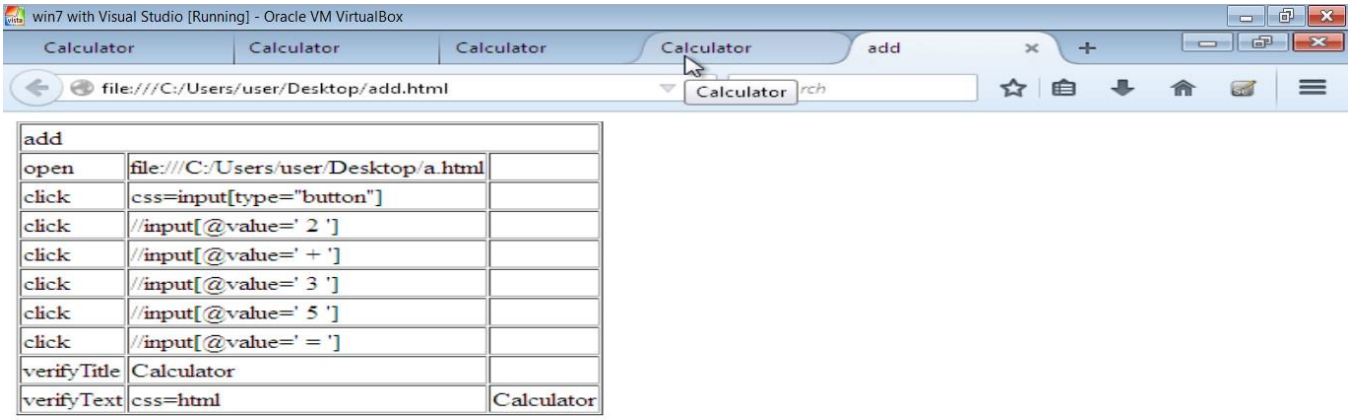
***** HTML Program For Calculator *****

```
<html>
<head>
<title>Calculator </title>
</head>
 <body>
<div id='calc-contain'>
<form name="calculator">
<input type="text" name="answer" />
<br>
<input type="button" value=" 1 " onclick="calculator.answer.value += '1'" />
 <input type="button" value=" 2 " onclick="calculator.answer.value += '2'" />
<input type="button" value=" 3 " onclick="calculator.answer.value += '3'" />
<input type="button" value=" + " onclick="calculator.answer.value += '+'" />
<br/>
<input type="button" value=" 4 " onclick="calculator.answer.value += '4'" />
 <input type="button" value=" 5 " onclick="calculator.answer.value += '5'" />
<input type="button" value=" 6 " onclick="calculator.answer.value += '6'" />
<input type="button" value=" - " onclick="calculator.answer.value += '-'" />
</br>
<input type="button" value=" 7 " onclick="calculator.answer.value += '7'" />
<input type="button" value=" 8 " onclick="calculator.answer.value += '8'" />
<input type="button" value=" 9 " onclick="calculator.answer.value += '9'" />
<input type="button" value=" x " onclick="calculator.answer.value += '*'" />
</br>
<input type="button" value=" c " onclick="calculator.answer.value = "" />
<input type="button" value=" 0 " onclick="calculator.answer.value += '0'" />
<input type="button" value=" = " onclick="calculator.answer.value =
eval(calculator.answer.value)" />
 <input type="button" value=" / " onclick="calculator.answer.value += '/'" />
</br>
```

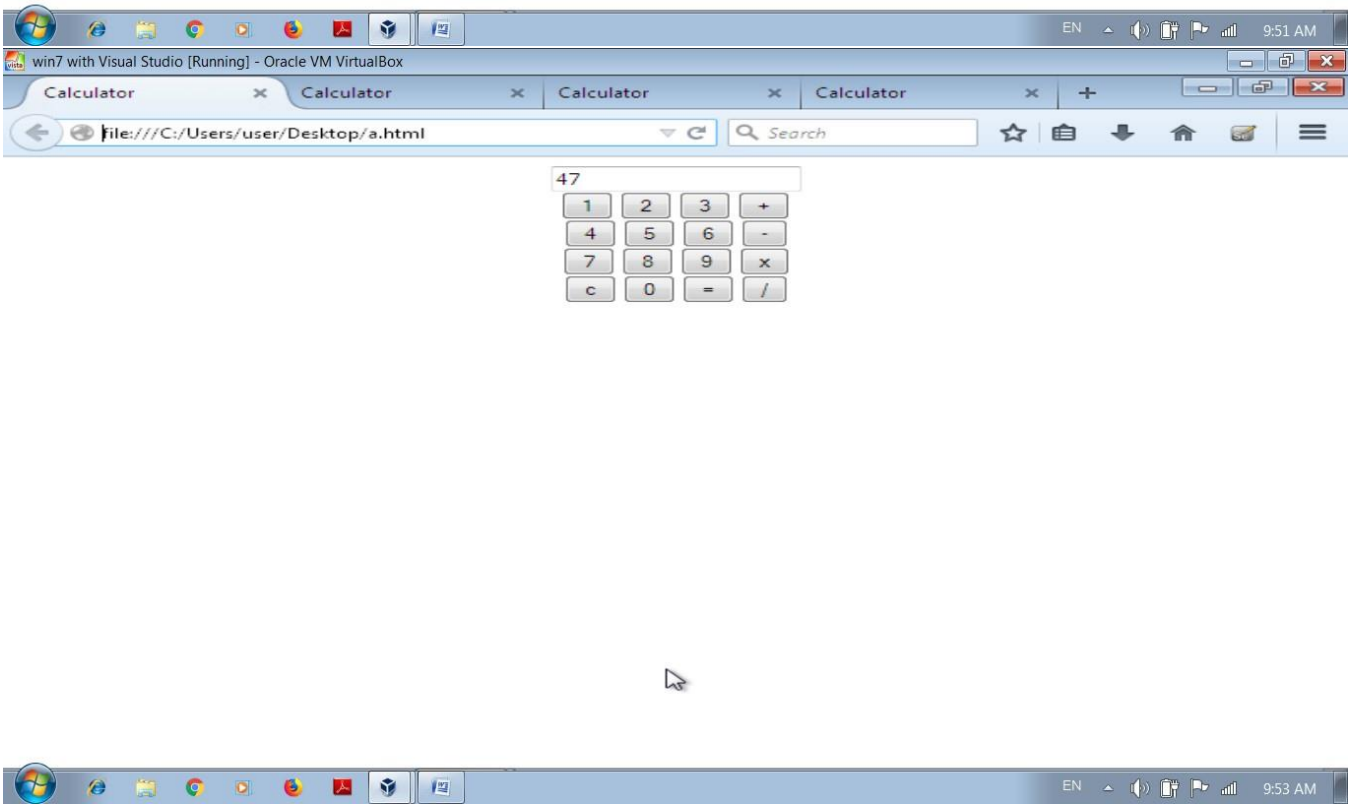
```
 </form>
<div id="agh">
 <p>
 </div>
 </div>
</body>
</html>
```



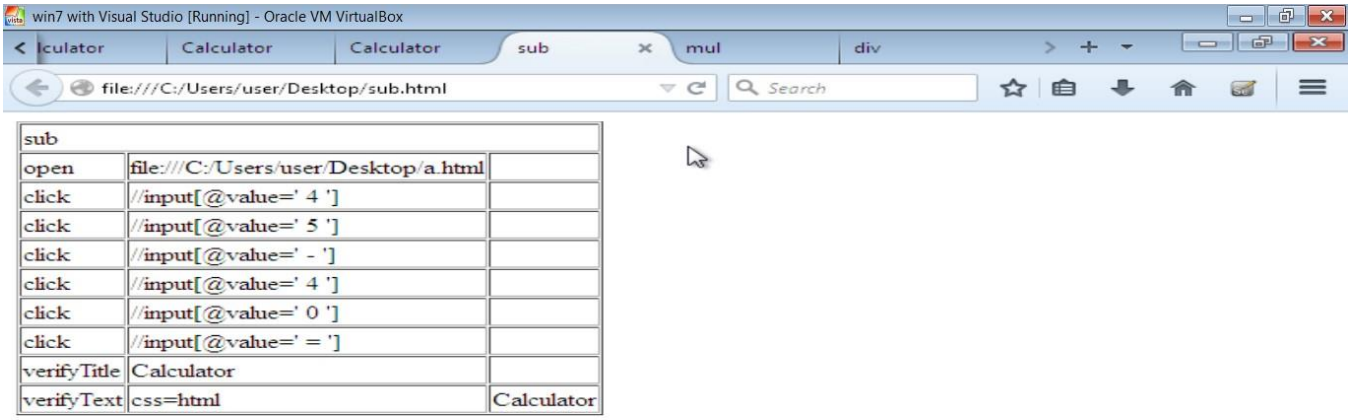
Test Case no. 1 – Add



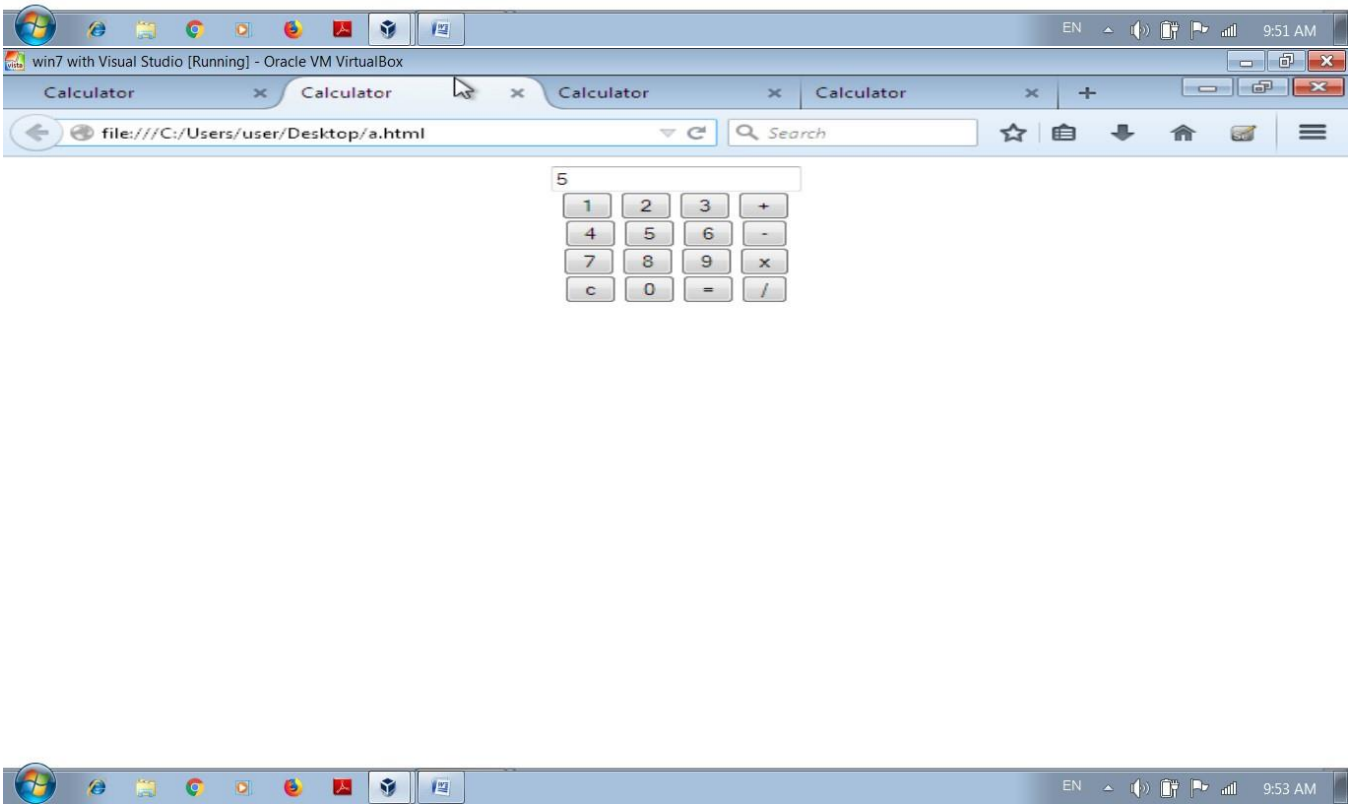
add		
open	file:///C:/Users/user/Desktop/a.html	
click	css=input[type="button"]	
click	//input[@value=' 2 ']	
click	//input[@value=' + ']	
click	//input[@value=' 3 ']	
click	//input[@value=' 5 ']	
click	//input[@value=' = ']	
verifyTitle	Calculator	
verifyText	css=html	Calculator



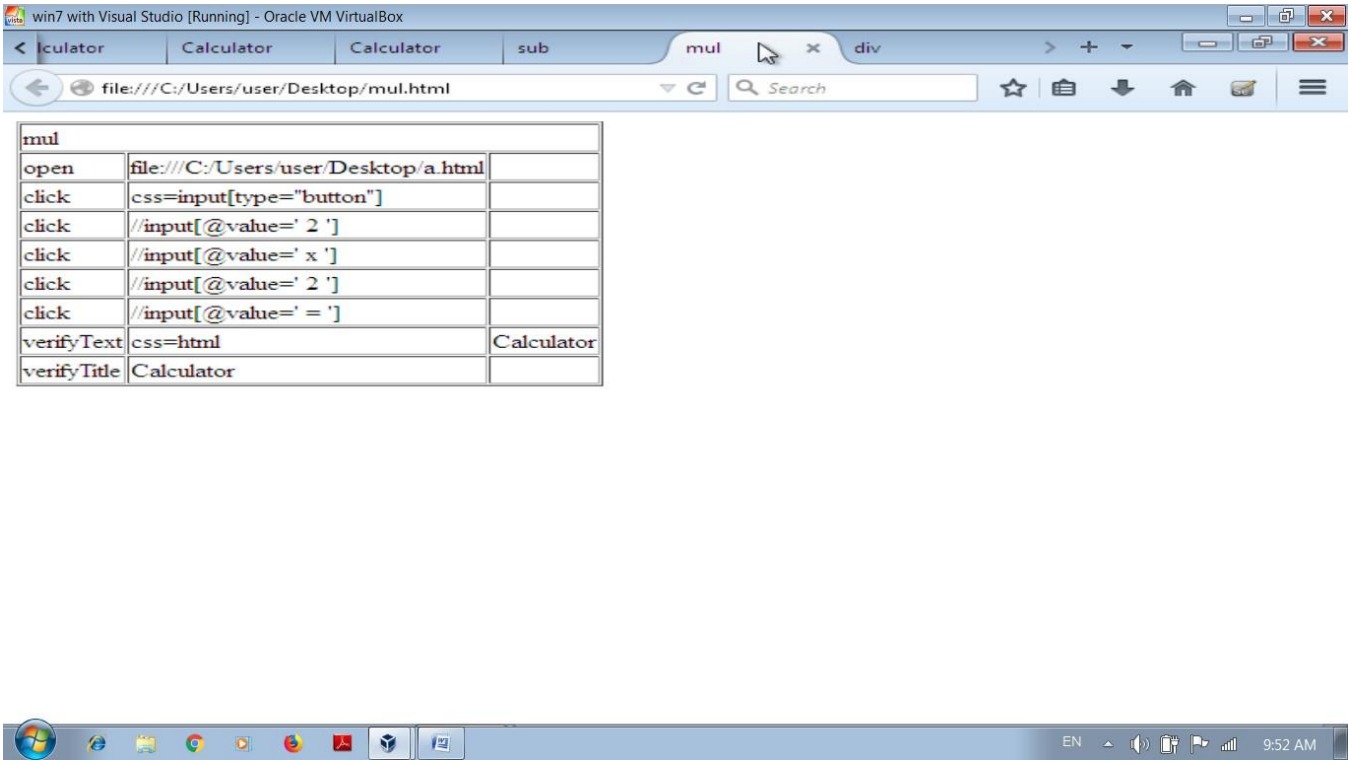
Test Case no. 2 - Sub



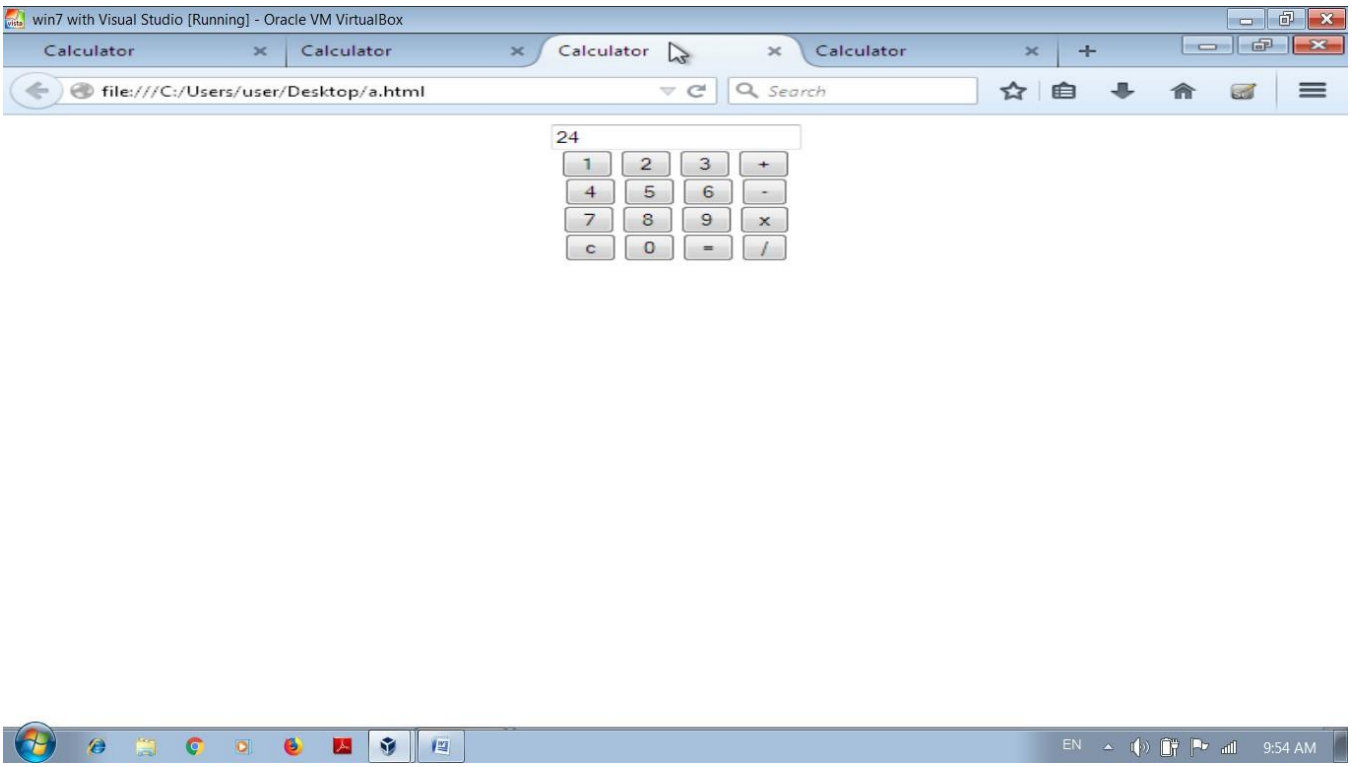
sub		
open	file:///C:/Users/user/Desktop/a.html	
click	//input[@value=' 4 ']	
click	//input[@value=' 5 ']	
click	//input[@value=' - ']	
click	//input[@value=' 4 ']	
click	//input[@value=' 0 ']	
click	//input[@value=' = ']	
verifyTitle	Calculator	
verifyText	css=html	Calculator



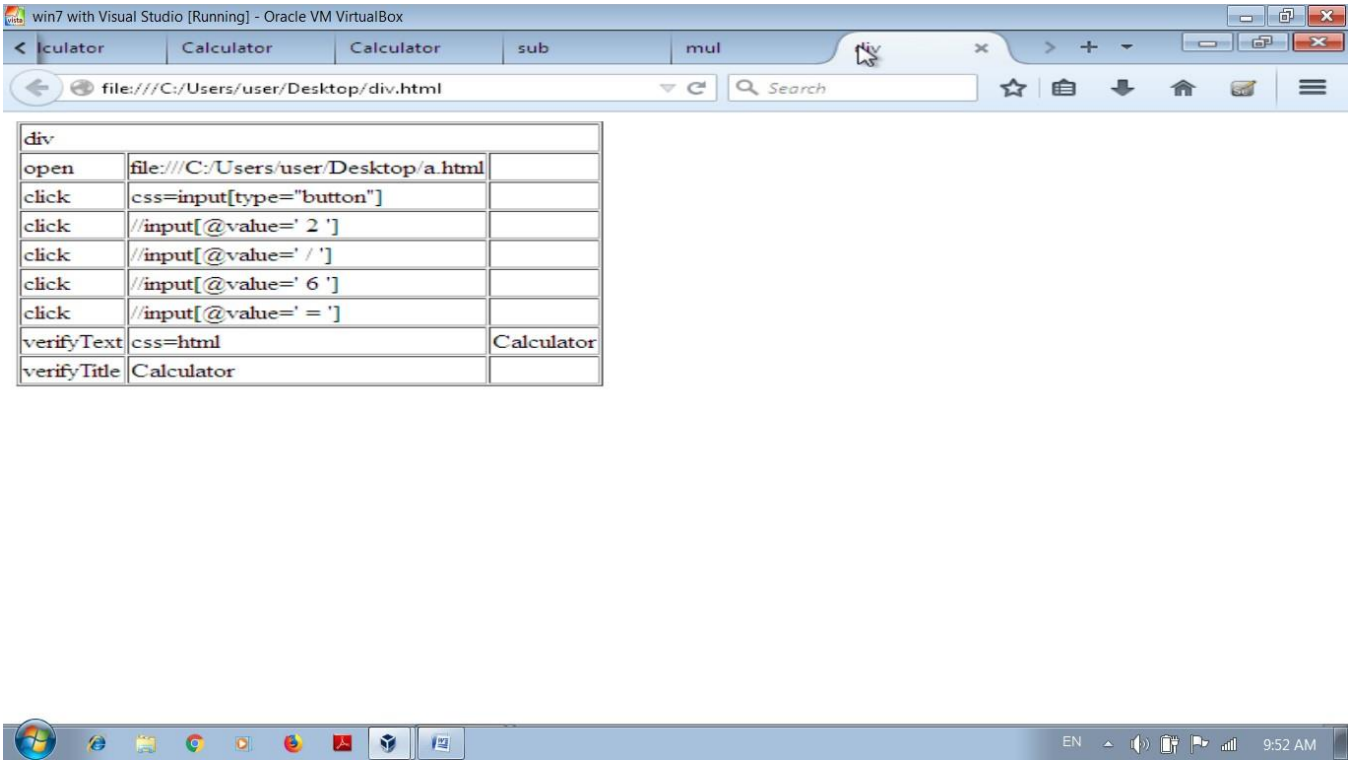
Test Case no .3 - Mul



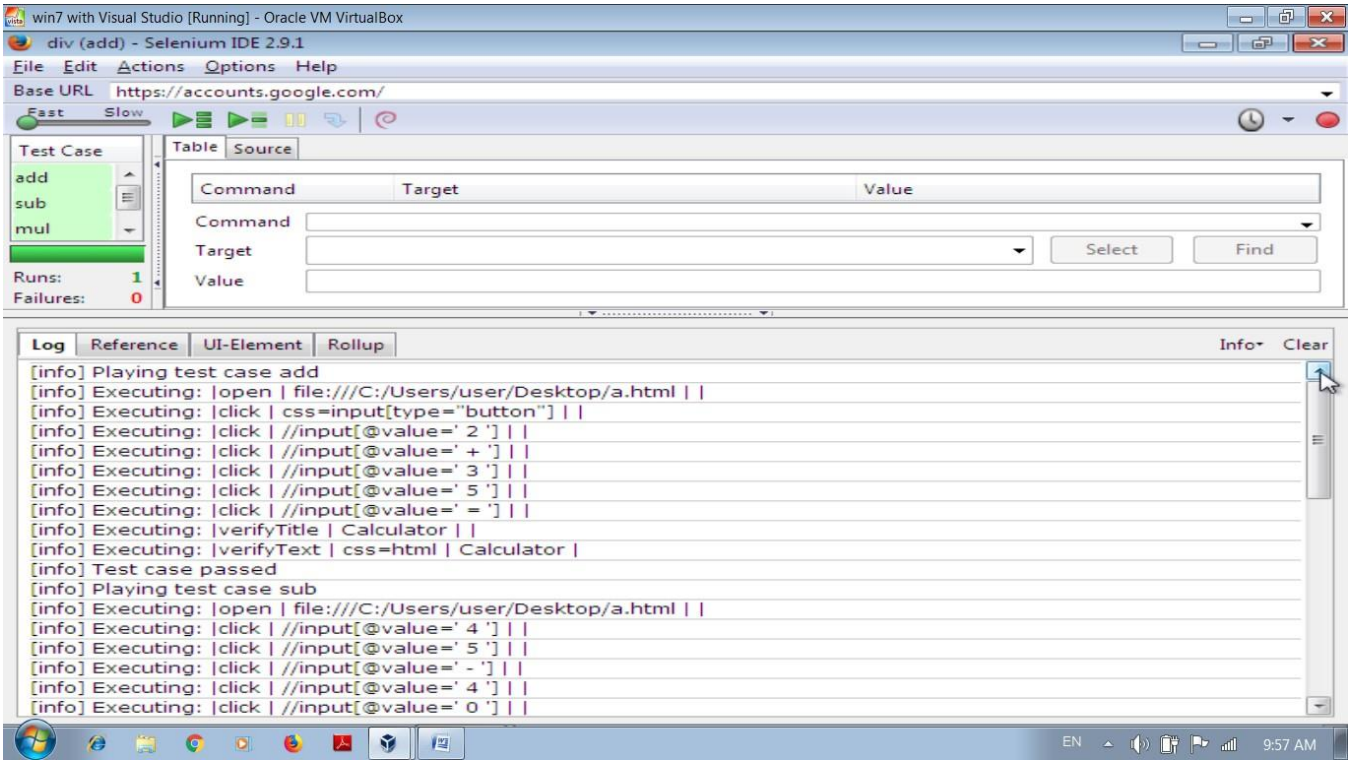
mul		
open	file:///C:/Users/user/Desktop/a.html	
click	css=input[type="button"]	
click	//input[@value=' 2 ']	
click	//input[@value=' x ']	
click	//input[@value=' 2 ']	
click	//input[@value=' = ']	
verifyText	css=html	Calculator
verifyTitle	Calculator	

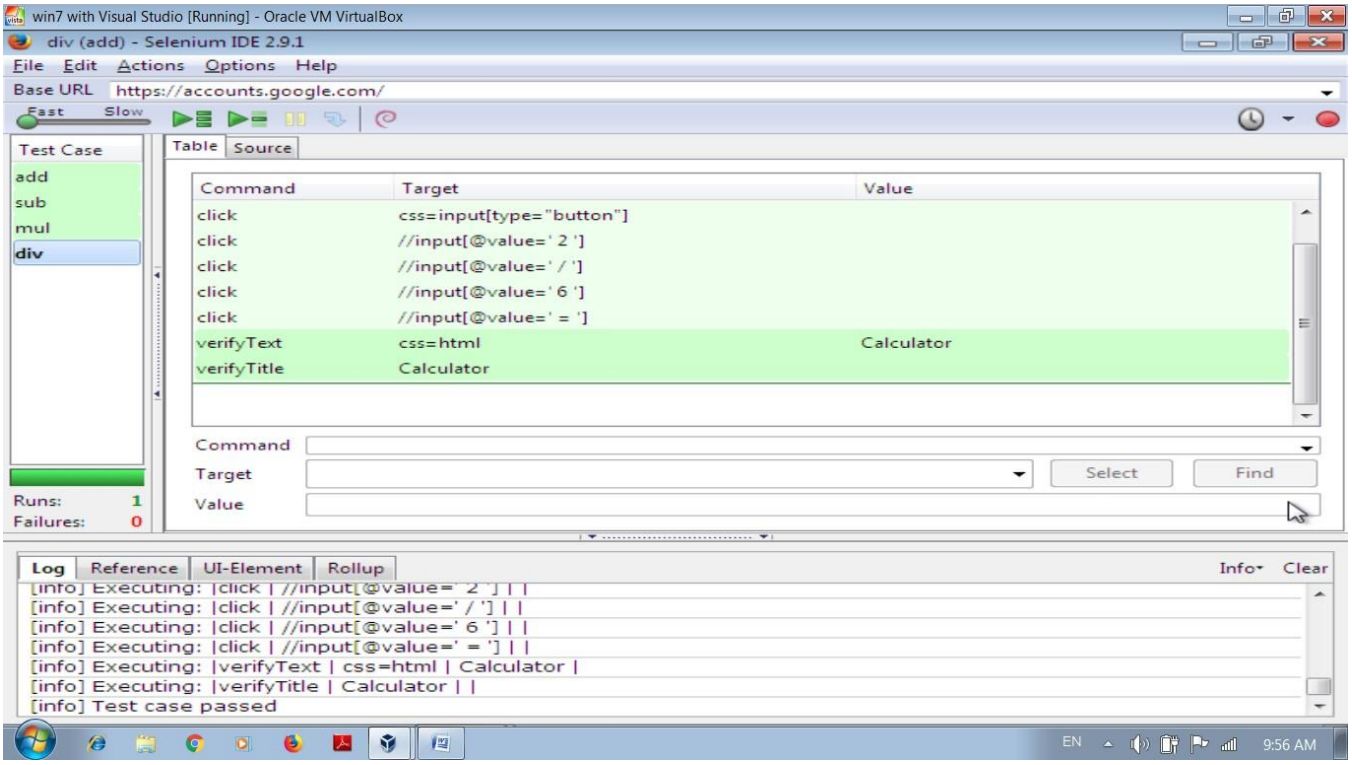


Test Case no .4 -Div



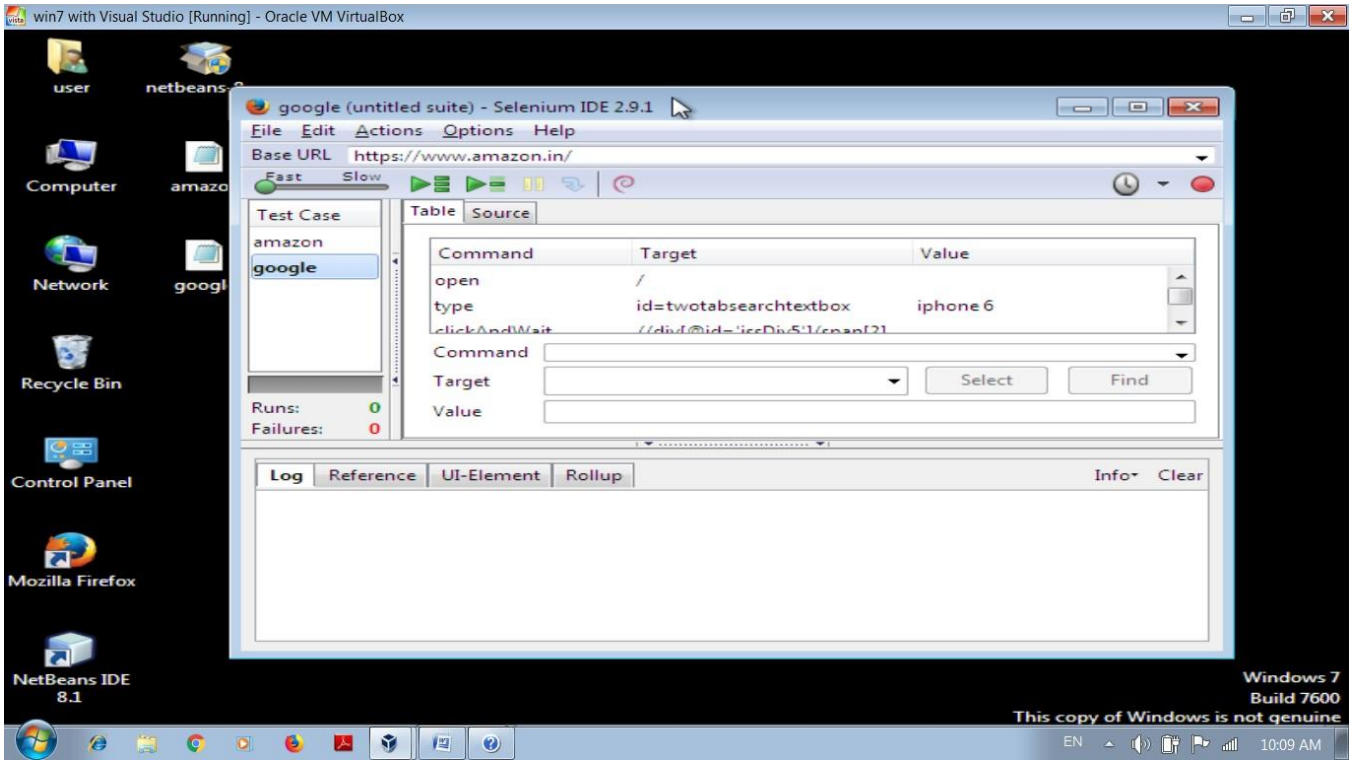
Result of All Test Cases (Output):-



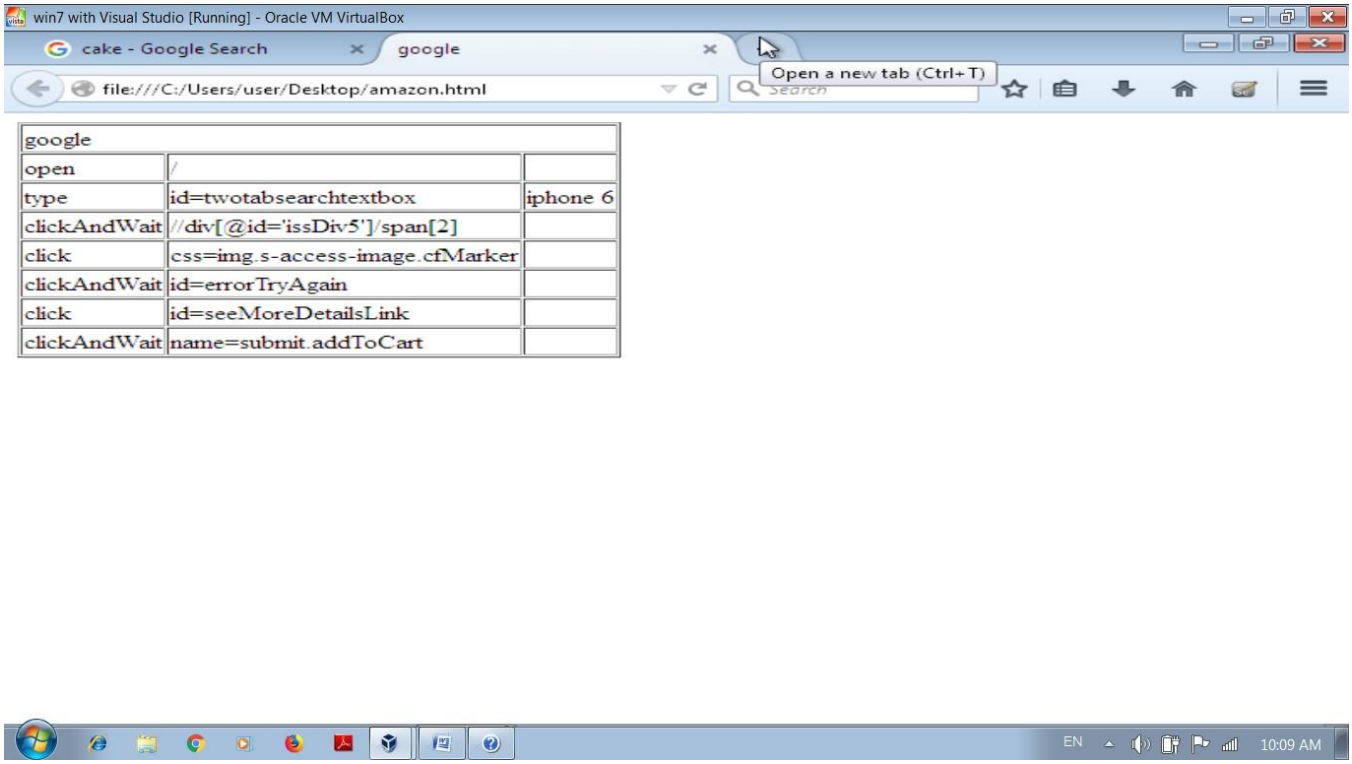


Practical No :- 2

Aim:- Conduct a test suite for any two web sites.

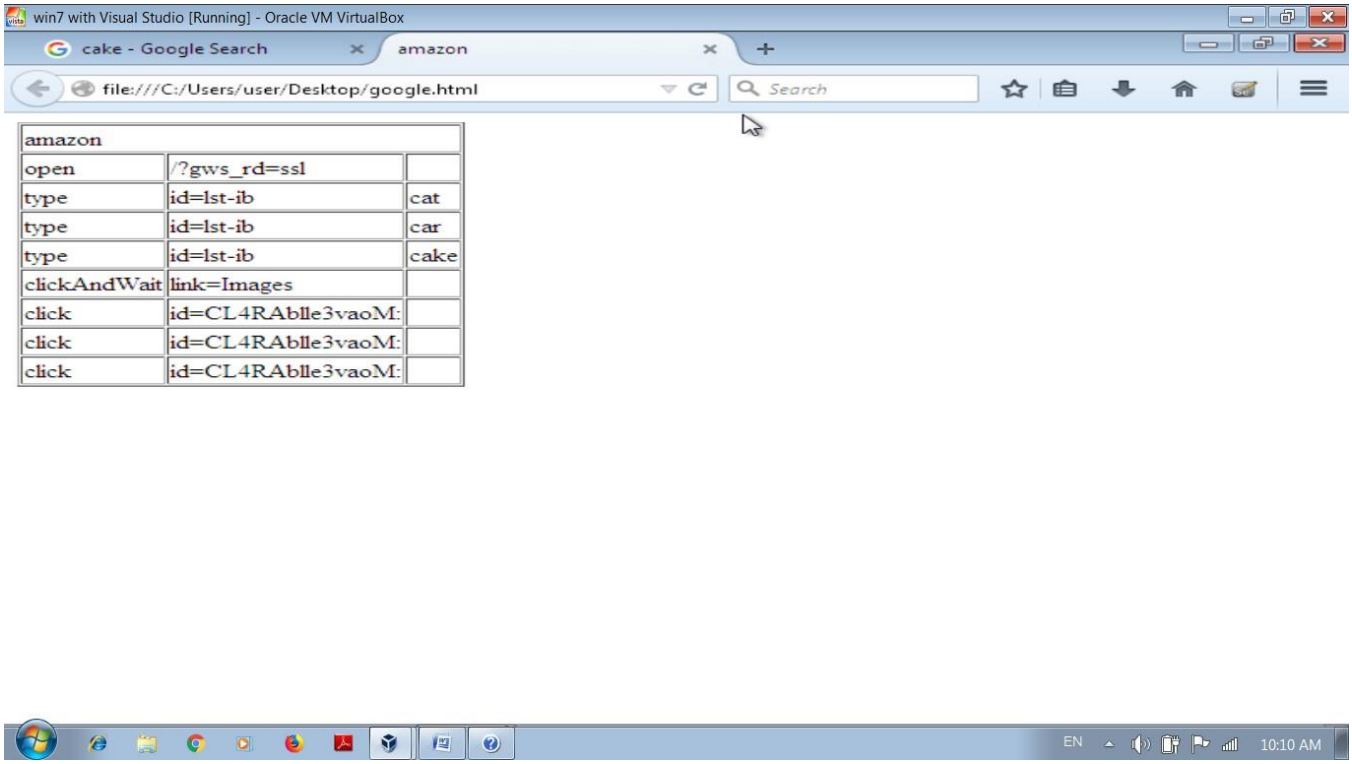


Test Case 1 :- www.amazon.in



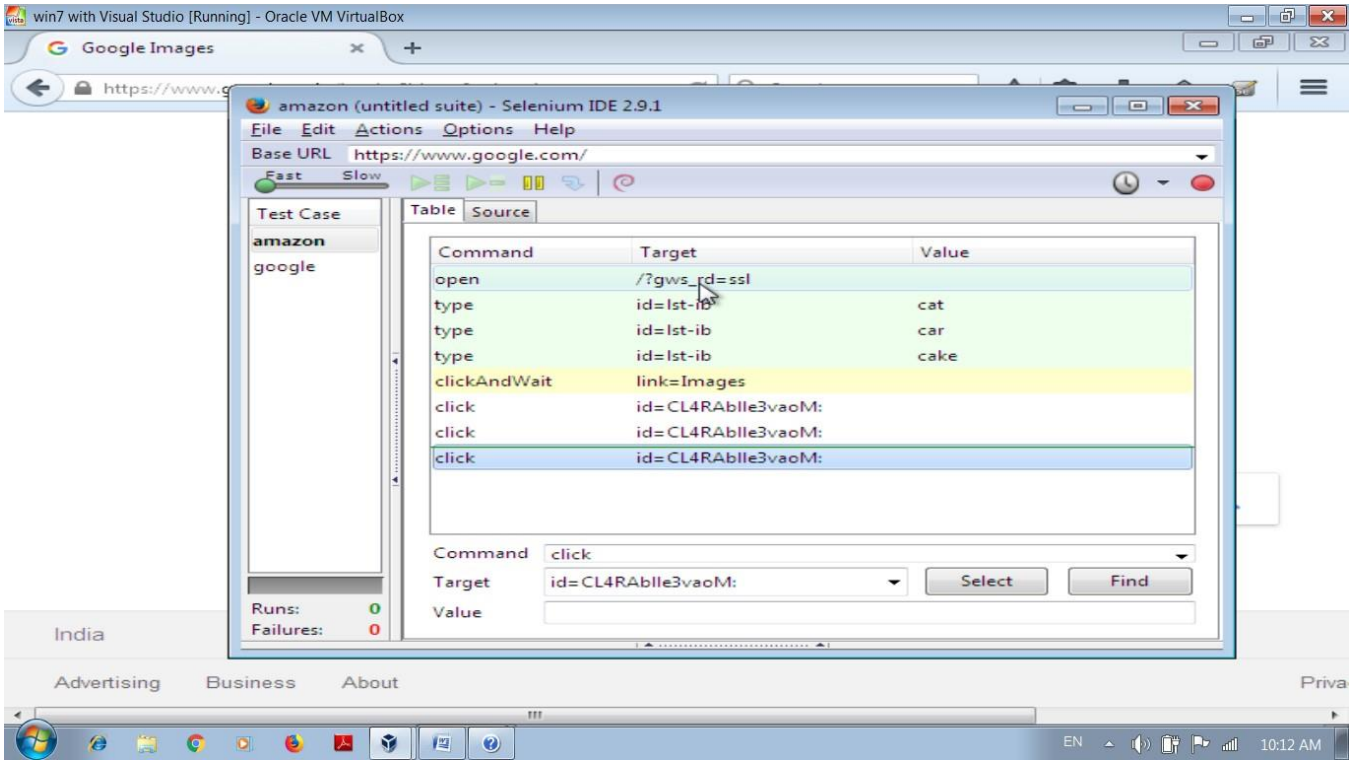
google		
open	/	
type	id=twotabsearchtextbox	iphone 6
clickAndWait	//div[@id='issDiv5']/span[2]	
click	css=img.s-access-image.cfMarker	
clickAndWait	id=errorTryAgain	
click	id=seeMoreDetailsLink	
clickAndWait	name=submit.addToCart	

Test Case 2 :- www.google.com (surfing)

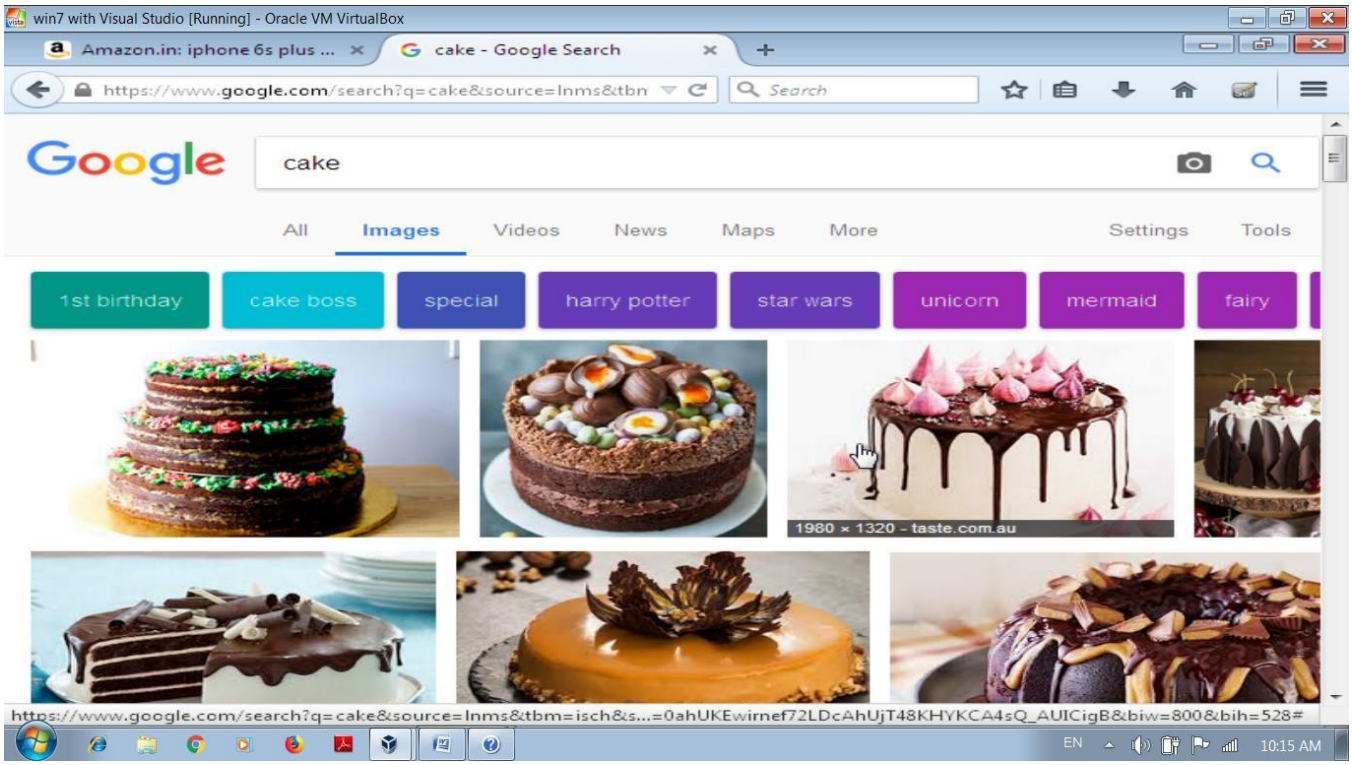


amazon		
open	/?gws_rd=ssl	
type	id=lst-ib	cat
type	id=lst-ib	car
type	id=lst-ib	cake
clickAndWait	link=Images	
click	id=CL4RAblle3vaoM:	
click	id=CL4RAblle3vaoM:	
click	id=CL4RAblle3vaoM:	

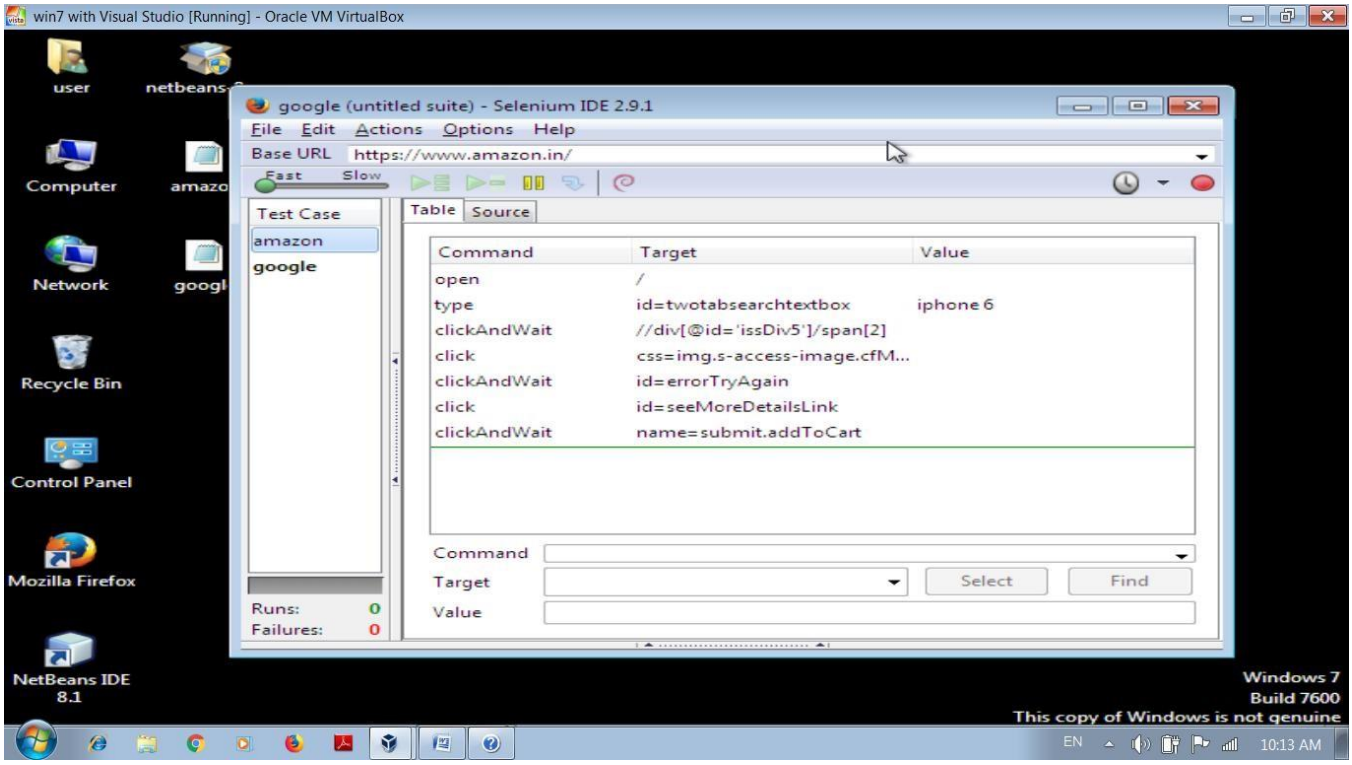
www.Google.com activity recorded.



Output of Running Test :-



www.amazon.in Activity recorded.



Output of Running Test Case :-

Practical No :- 3

Aim:- Install Selenium server (Selenium RC) and demonstrate it using a script in
Java / PHP.

Steps:-

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.

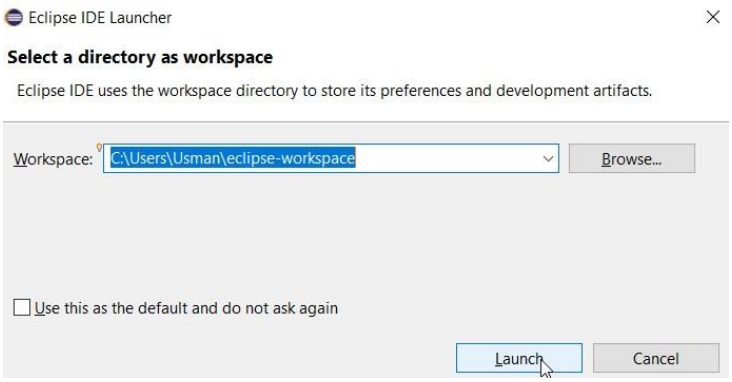
We have finished setting up the eclipse.

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)

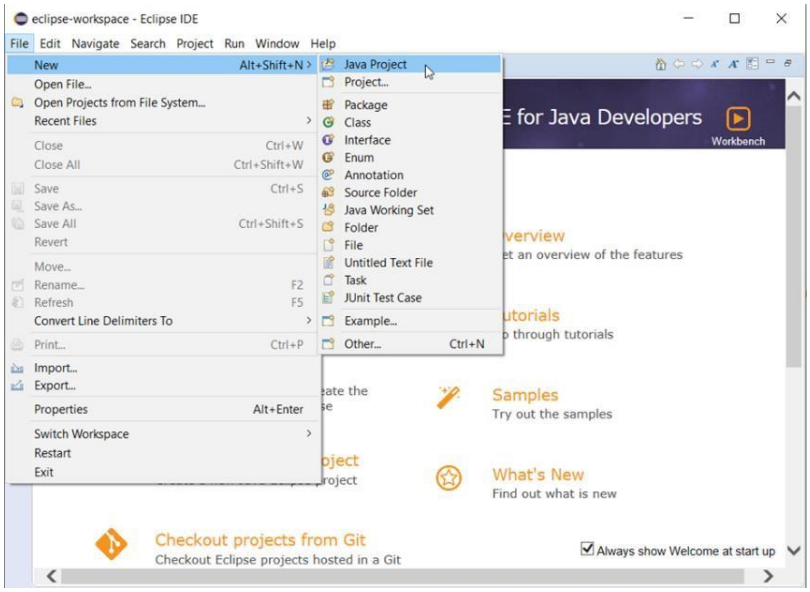
Downloading and unzipping the files into a folder is done.
We need to configure the appropriate Selenium Client driver Jar file to the Eclipse.

1) Open Eclipse. Select your appropriate workspace directory. Click Launch:-



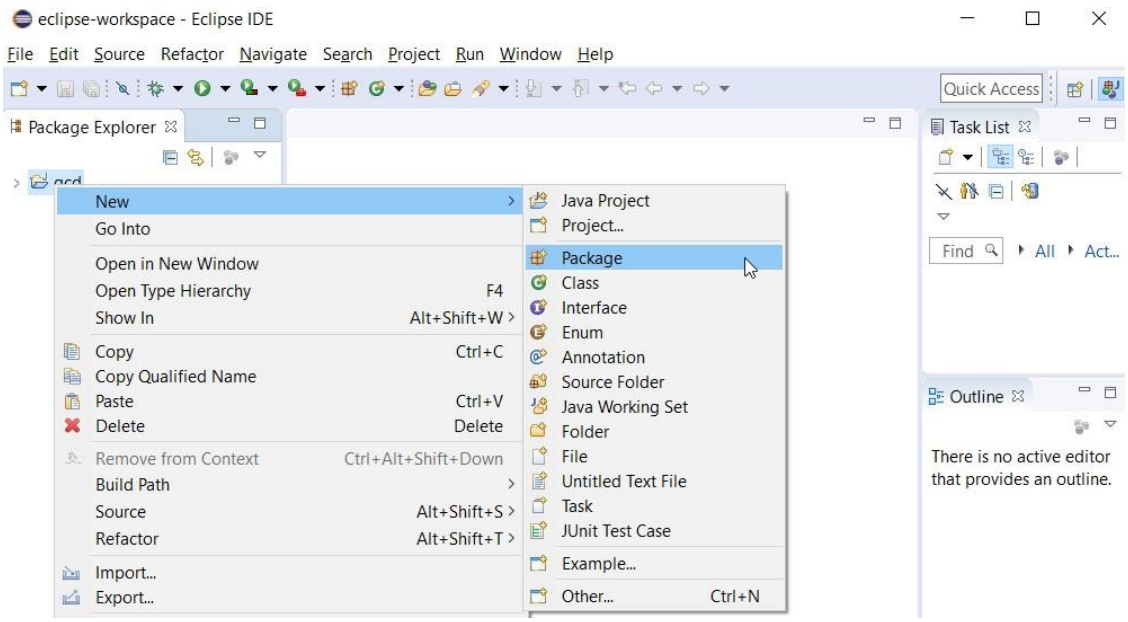
2) Create a Project (File > New > Java Project) and give the project name as “simple_
program”

3) Click Finish.



Now we are done with creation of project and need to configure the Selenium
Client driver to this Project.

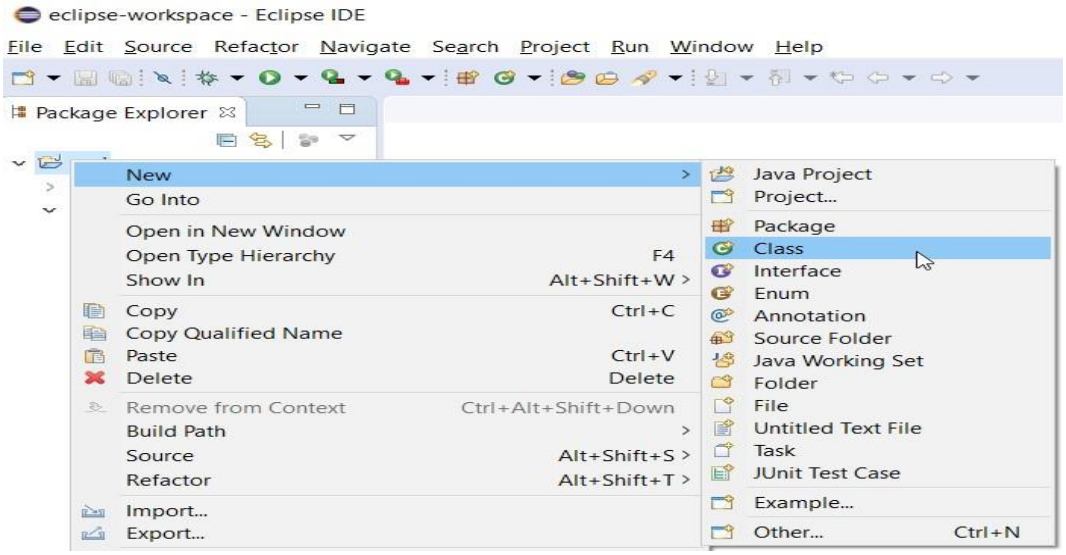
4) Create a Package (right-click on Project Name > New > Package > Name it as “test”
5) Click Finish.



Steps for configuring the Selenium Client driver & creation of package and class file to this Project

1. Right Click “simple_program” project
2. Click “Java Build Path”
3. Click Libraries tab
4. Click “Add External JARs” button
5. Select “Selenium Client Drivers” unzipped in C:Selenium folder (Selenium Server JAR file should not be added)
6. Click OK

7. Referenced libraries –> contains both the Selenium Client driver jar files.
8. Create a new class file as “test1” in the “simple_program” project by right click on src folder.

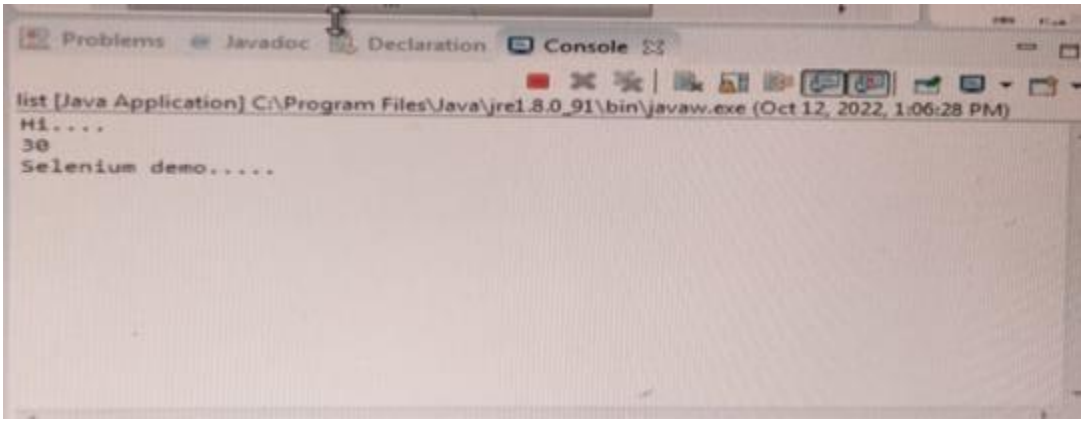


9. **Copy the below code in the class file:-**

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

public class test1
{
        static String driverPath = "D:\\selenium pracs\\geckodriver-
v0.21.0win32\\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();

      }
}
```

Output:-



Practical No :- 4

Aim :- Write and test a program to login a specific web page. (Using JUnit)

Steps:-

PART -1) Download JUnit using
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

Step 2) A "New System Variable" window will open:

3. Provide variable name as "JUNIT_HOME".

4. Provide JUnit value as JUnit path where you have copied JUnit jar files.
5. 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.

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.

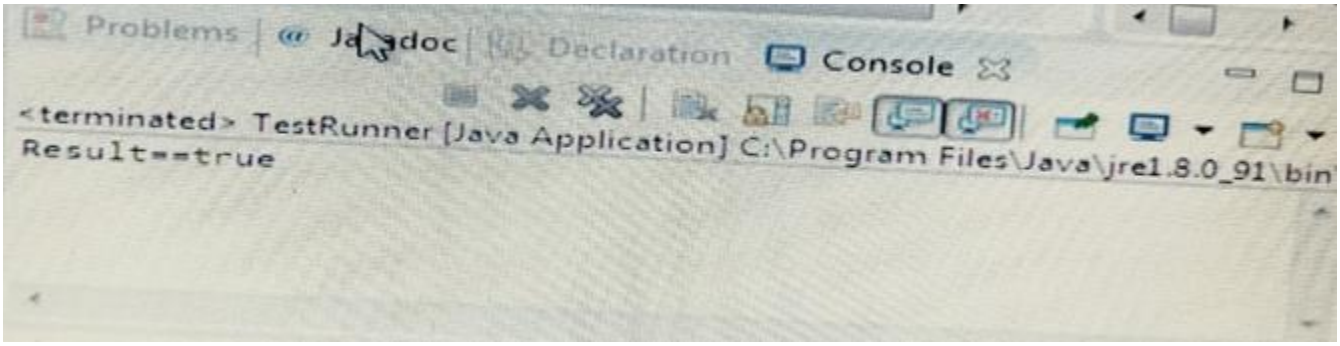
```
import org.junit.runner.JUnitCore; import
org.junit.runner.Result;
import org.junit.runner.notification.Failure;

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 "JUnit Test"

(Note:- If it is successfully executed , it will show a green check mark in front of it.)

Output:-



Practical No :- 5

Aim :- Write and test a program to update 6 students records into table into Excel file.

Steps for Installing TestNG and Launching Tests in eclipse:-

a) 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.

b) 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.

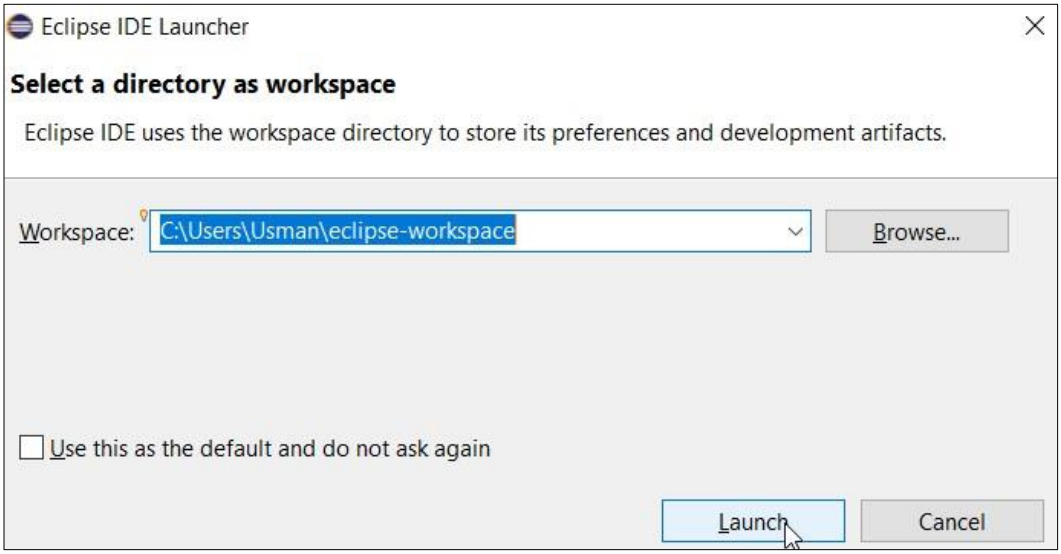
To access the methods and classes provided by this API inside Eclipse we need to add this JAR file to the Java Build Path.
(I have explained steps to add external Jar files to Java Build Path in previous examples)

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

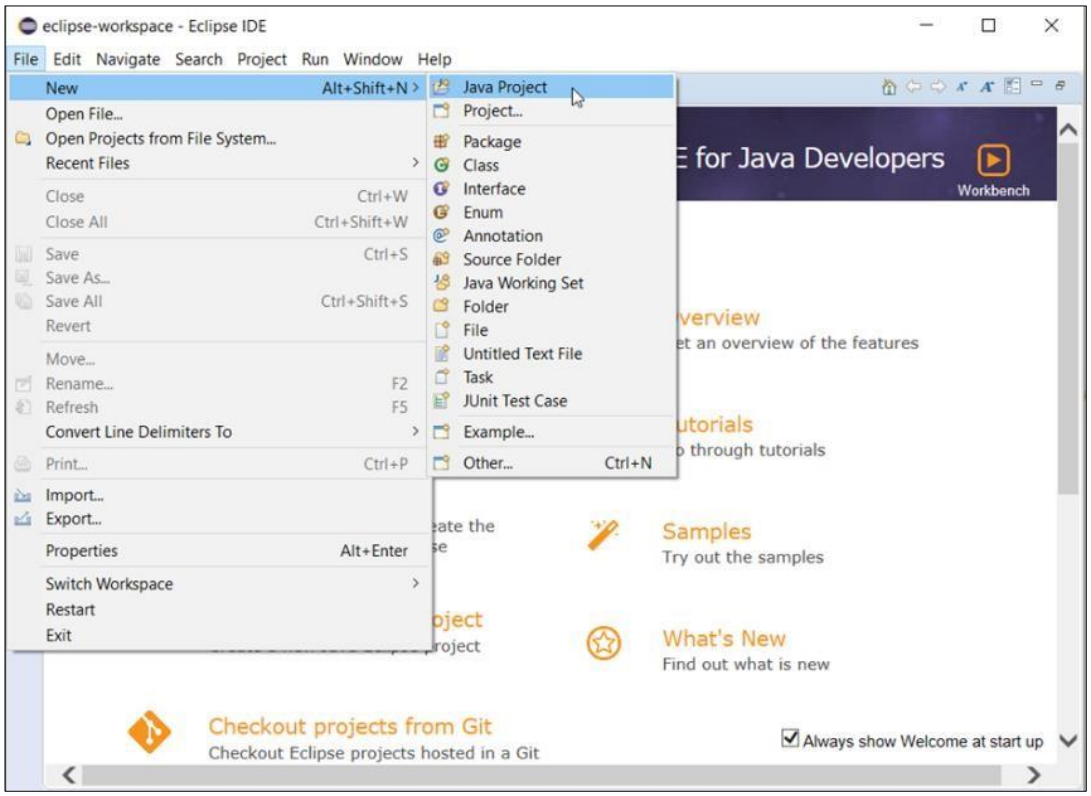
```
import jxl.Cell;
import jxl.Sheet;
import jxl.Workbook;
import jxl.read.biff.BiffException;
```

Steps for creating new project , package and class file :-

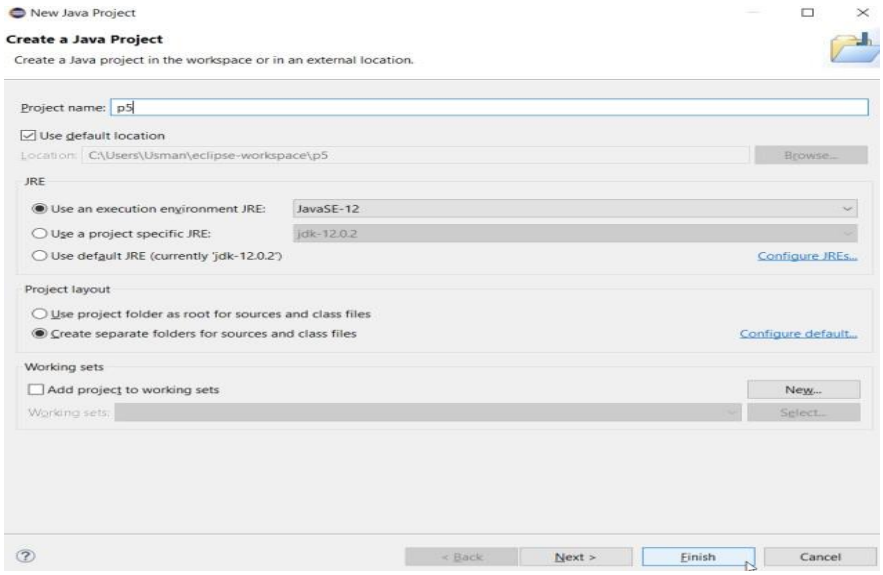
1) **Open Eclipse > Select your workspace directory > Click Launch.**



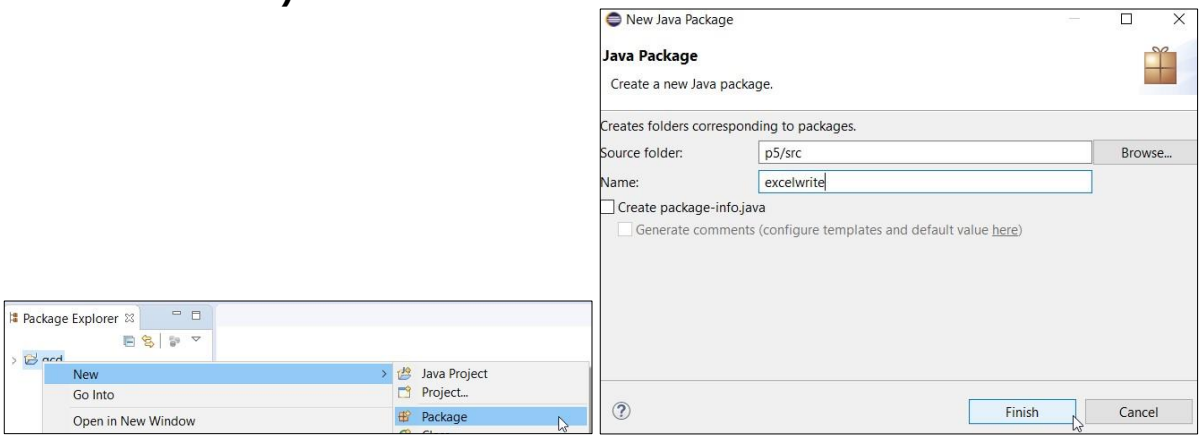
2) **Create a Project(File > New > Java Project):-**



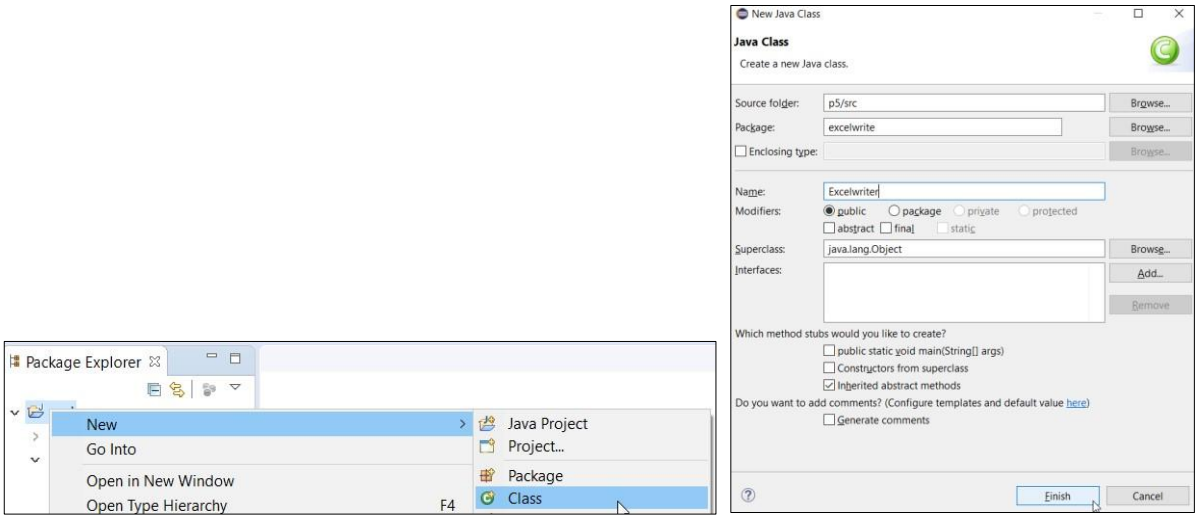
3) Name the project as “p5” > click Finish .



4) **Create a Package(right-click on Project Name > New > Package > Name it > Finish)**

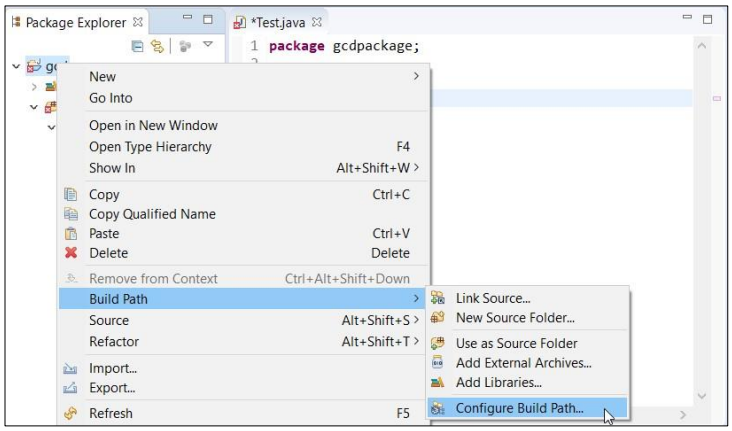


5) **Create a Class(right-click on Project Name > New > Class > Name it > Finish)**

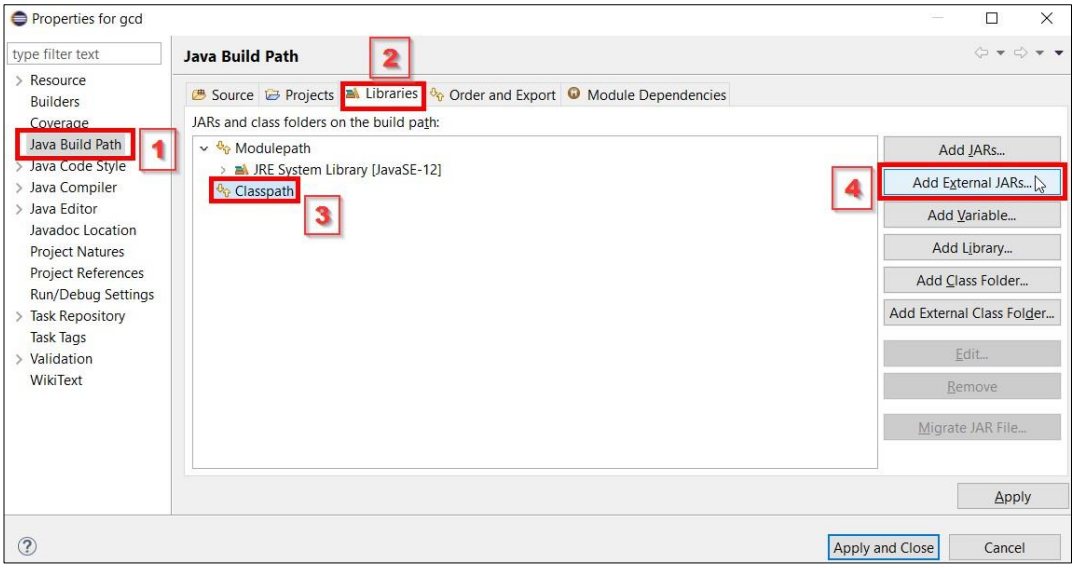


6) **Adding "JXL(JAR file)" in Eclipse IDE.**

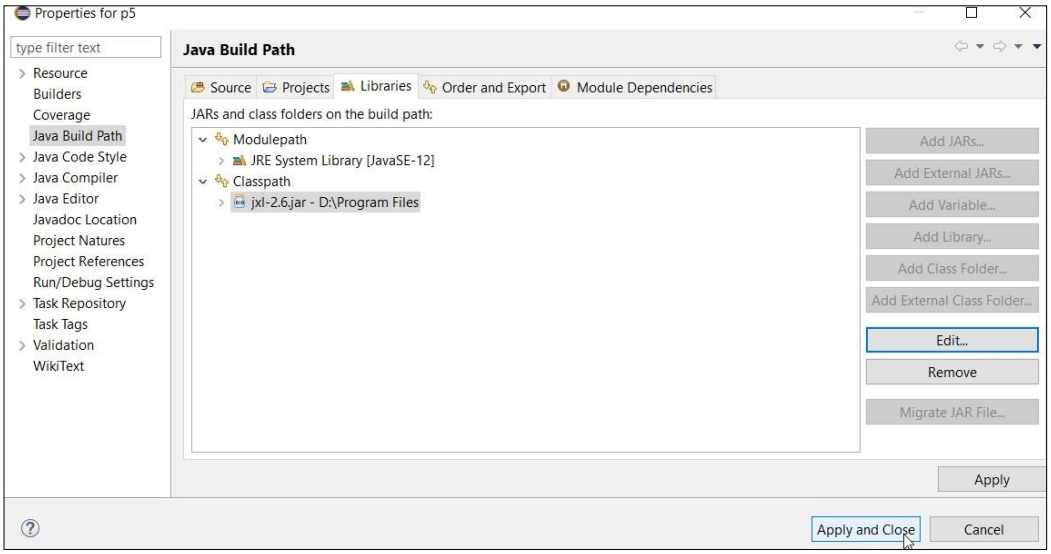
- **right-click on Project Name > Build Path > Configure Build Path.**



- **now go under: Java Build Path > Libraries > Classpath > click Add External JARs.**



- **Browse and add JAR file > click Apply and Close.**

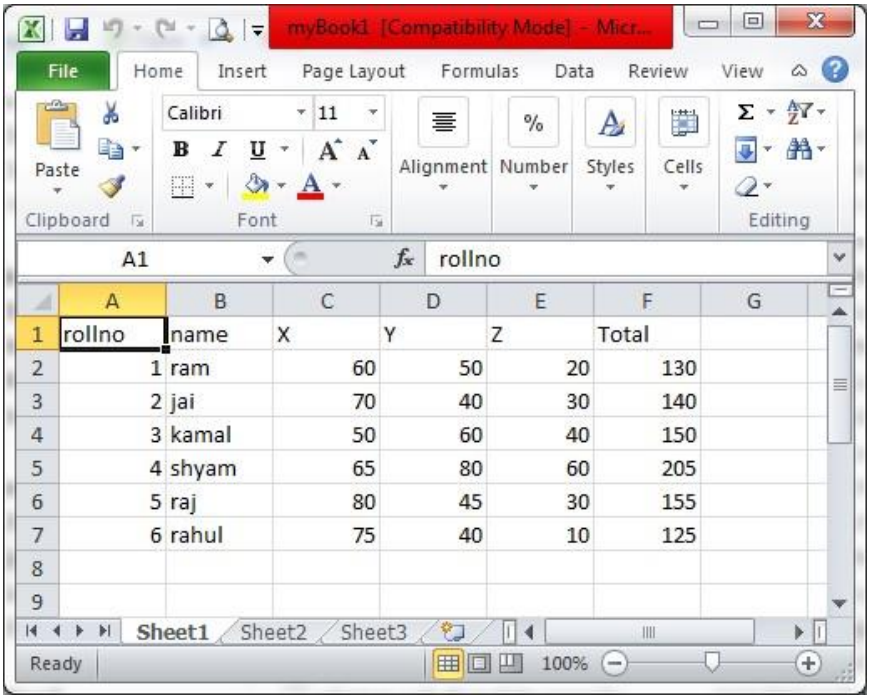


After this , Copy the below code in the class file:-

```
//Code to update 6 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\\Input");
       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\\Output");
```

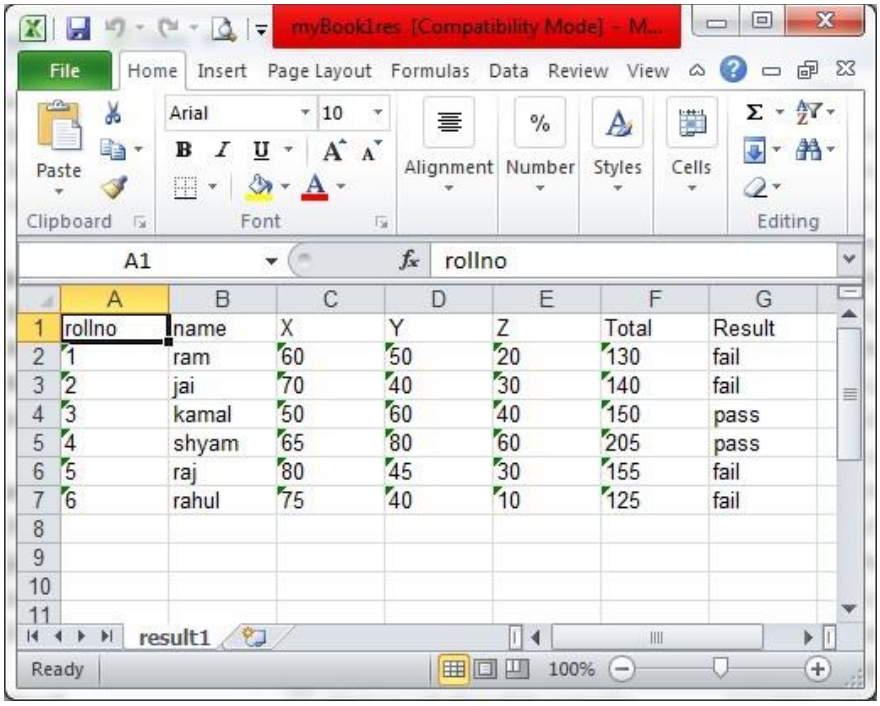
```
        WritableWorkbook wwb = Workbook.createWorkbook(fo);
         WritableSheet ws = wwb.createSheet("result1", 0);
        for (int i = 0; i < s.getRows(); i++)
         for (int j = 0; j < s.getColumns(); j++)
{
        a[i][j] = s.getCell(j, i).getContents();
        Label l2 = new Label(j, 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							

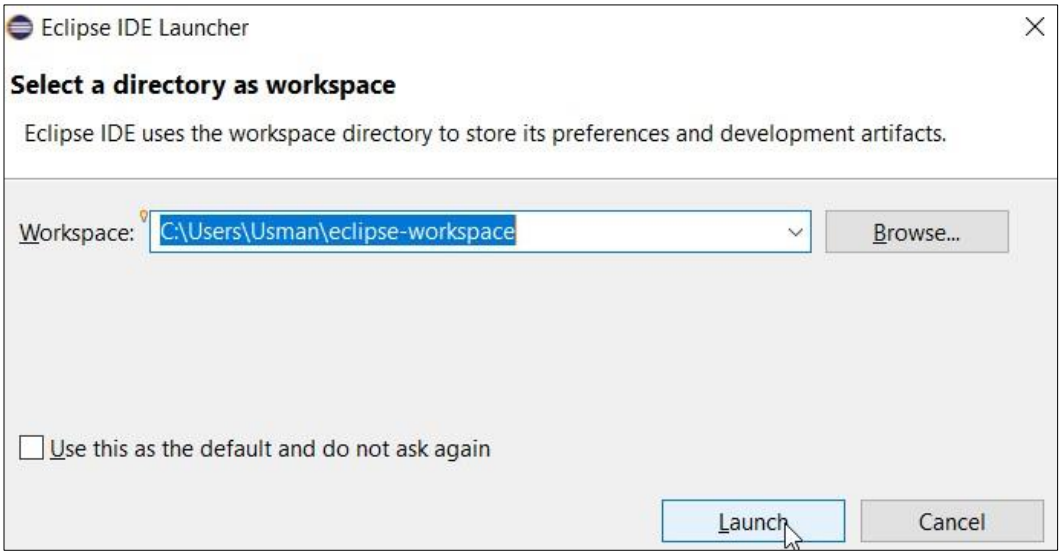
Practical No :- 6

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

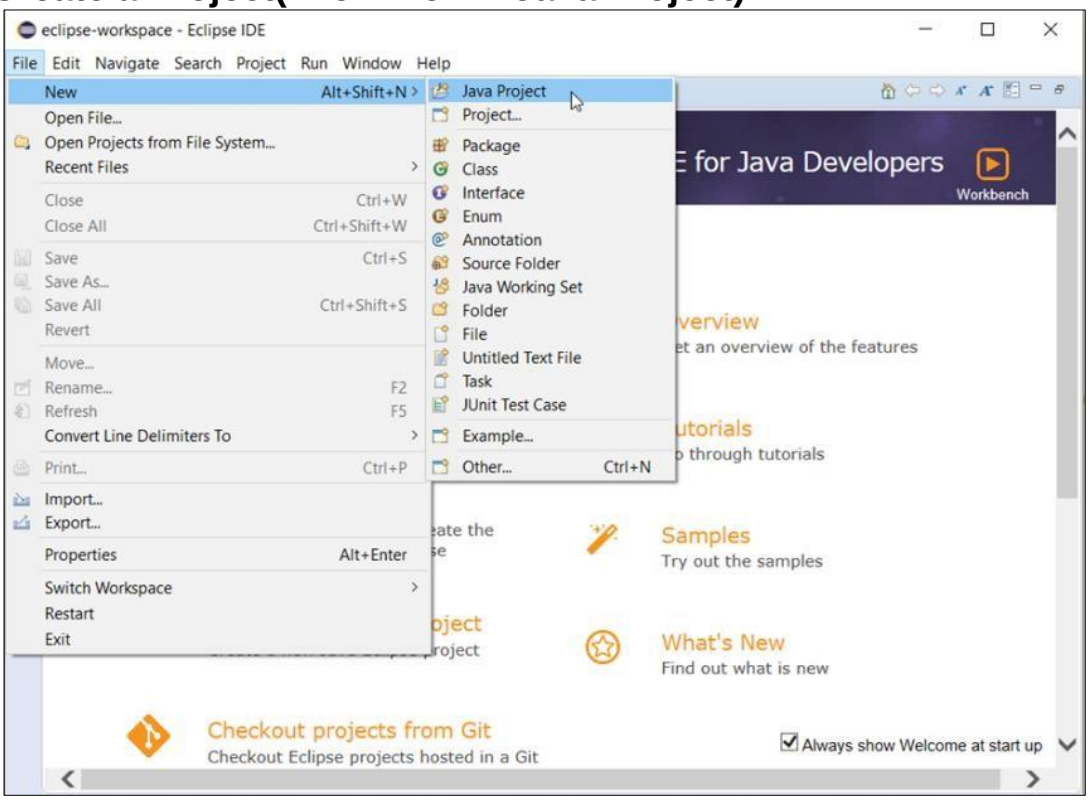
Steps:-

Steps for creating new project , package and class file :-

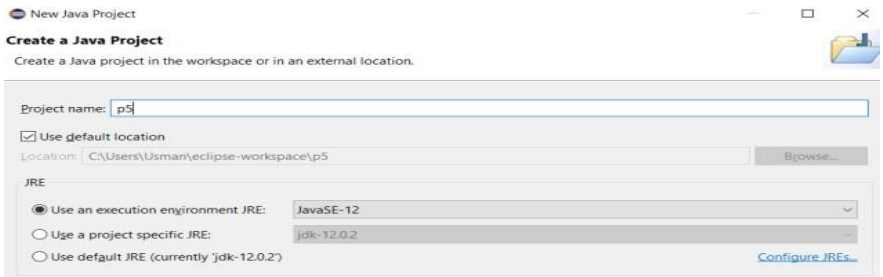
1) **Open Eclipse > Select your workspace directory > Click Launch.**



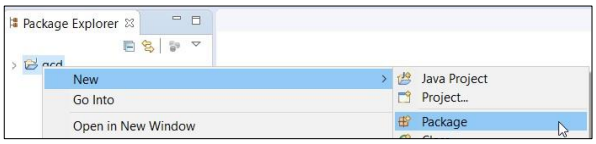
2) Create a Project(File > New > Java Project):-

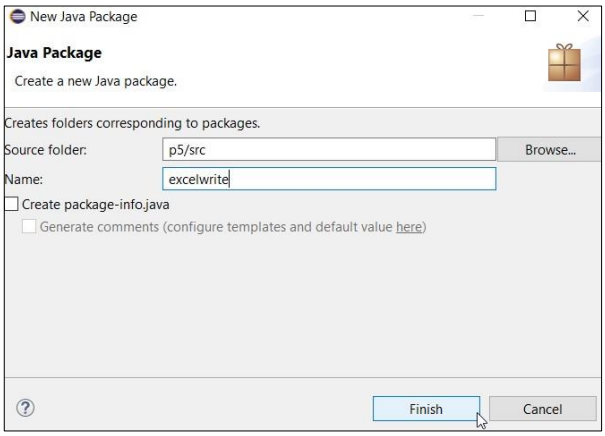


3) Name the project as “p5” > click Finish .

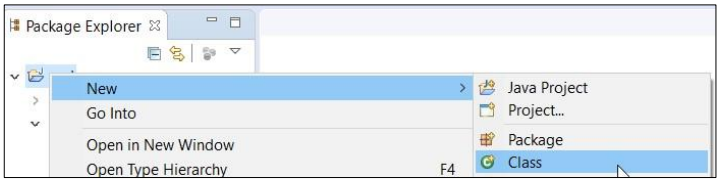


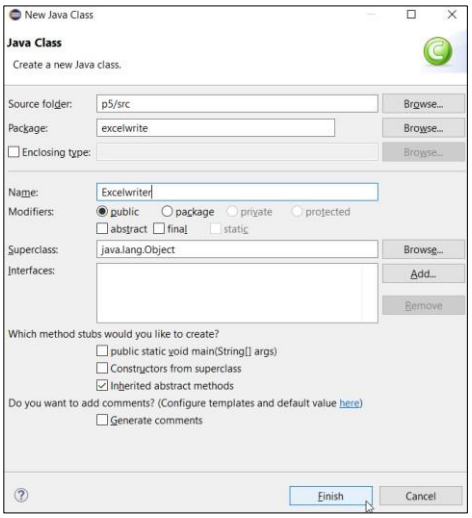
4) **Create a Package(right-click on Project Name > New > Package > Name it > Finish)**





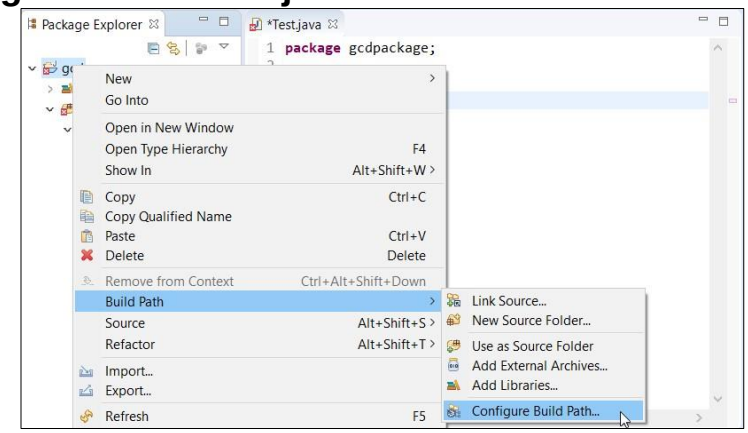
5) **Create a Class(right-click on Project Name > New > Class > Name it > Finish)**



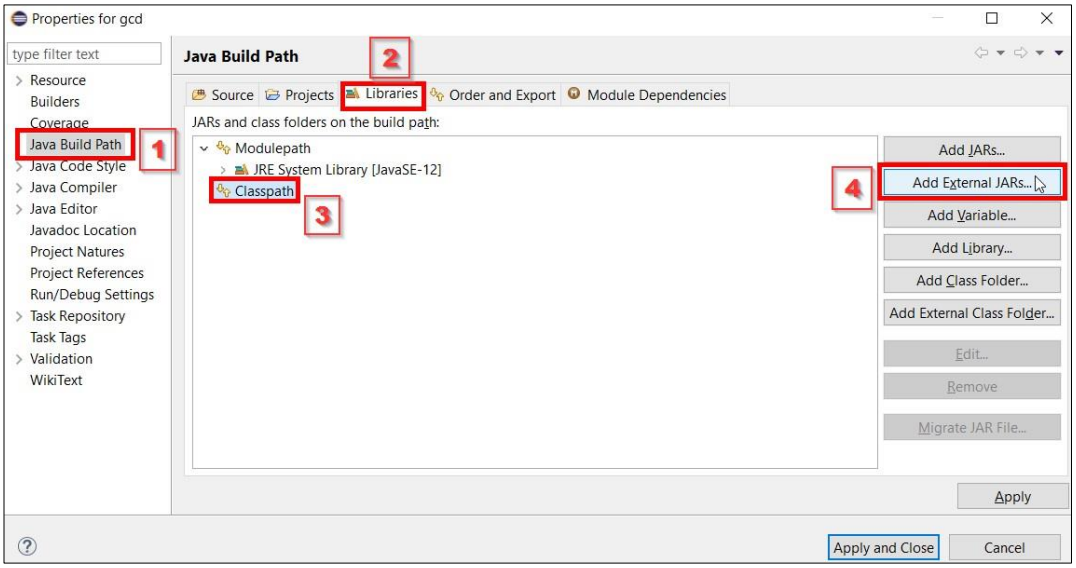


6) Adding "JXL(JAR file)" in Eclipse IDE.

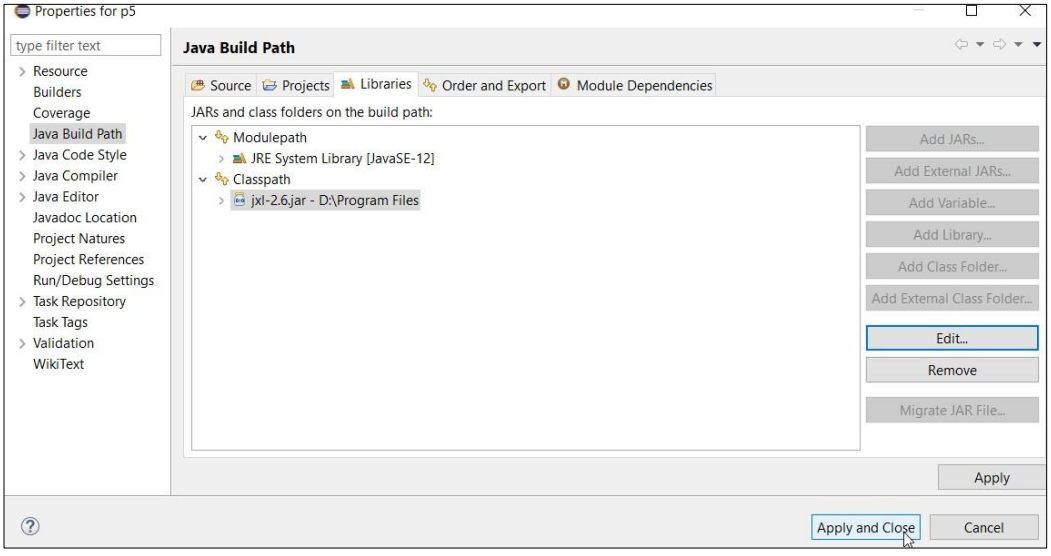
- **right-click on Project Name > Build Path > Configure Build Path.**



- **now go under: Java Build Path > Libraries > Classpath > click Add External JARs.**



- **Browse and add JAR file > click Apply and Close.**

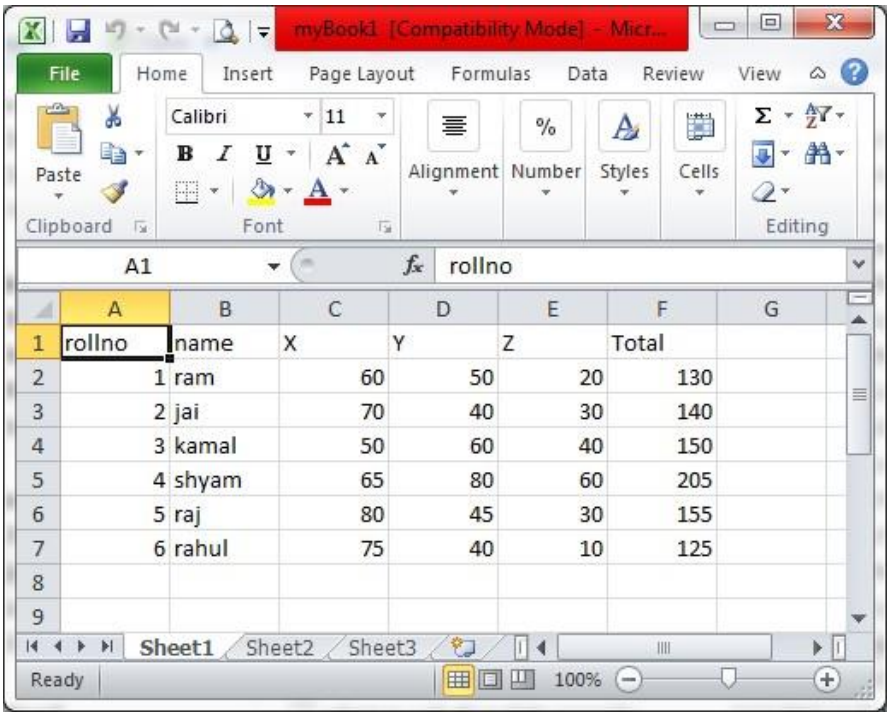


After this , Copy the below code in the class file:-

```
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\\Input");
      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\\Output");
        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();
```

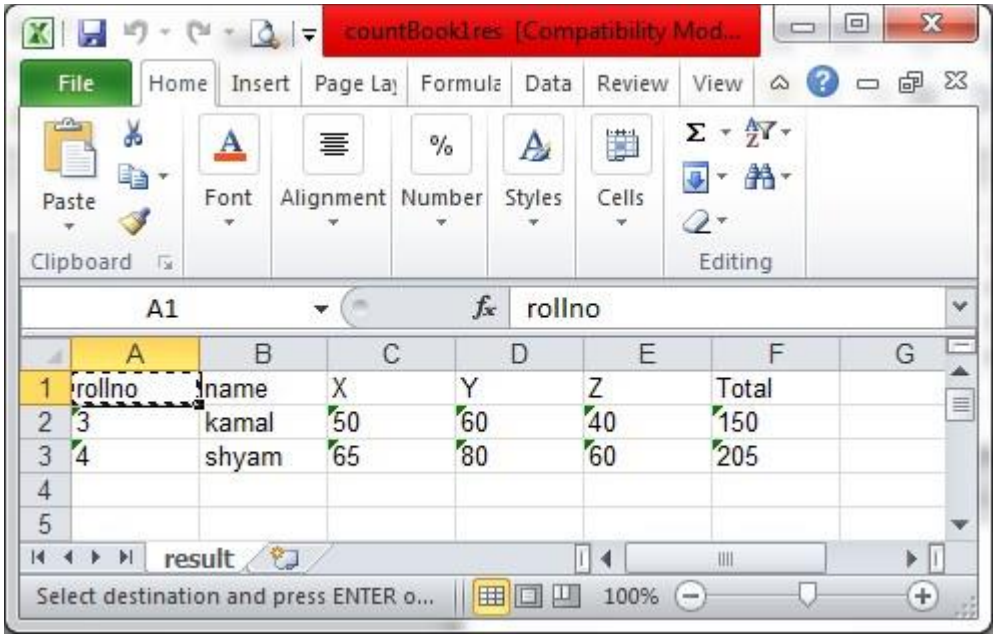
```
wwb.close();
  }
}
```

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:-

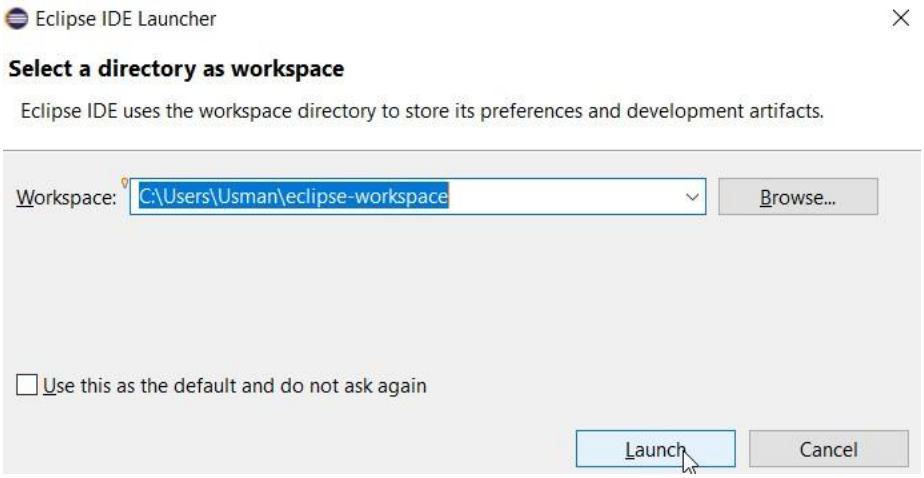


Practical No :- 7

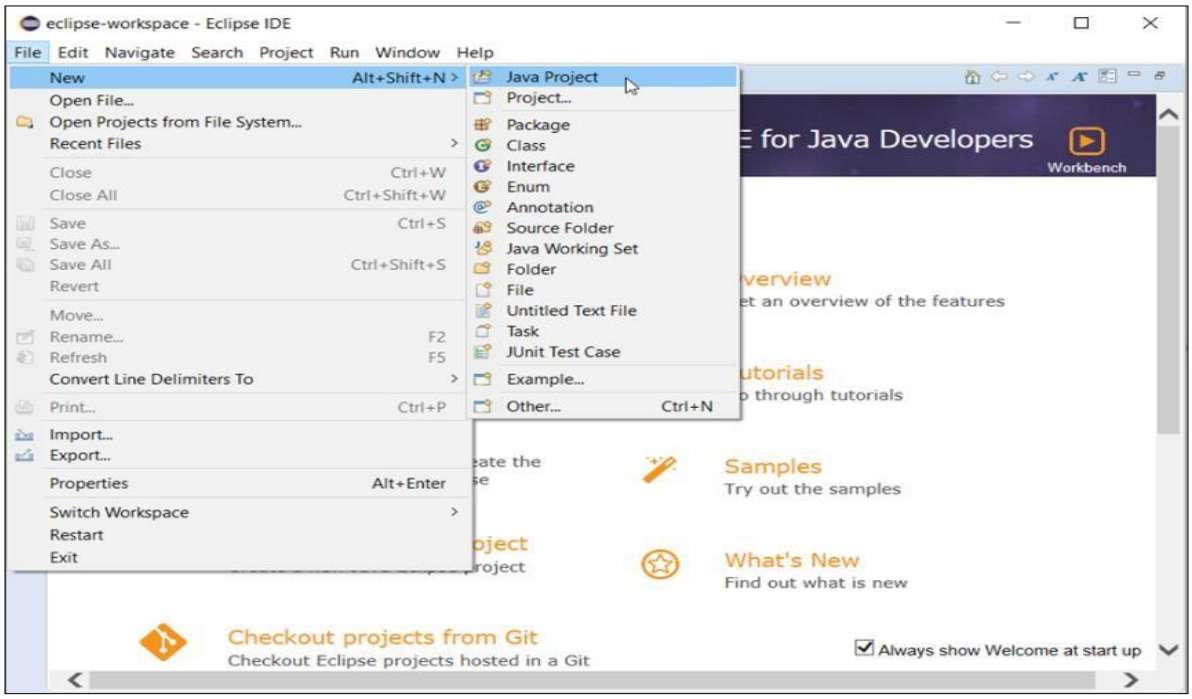
Aim :- **Write and test a program to get the number of objects present / available on the page.**

Steps:-

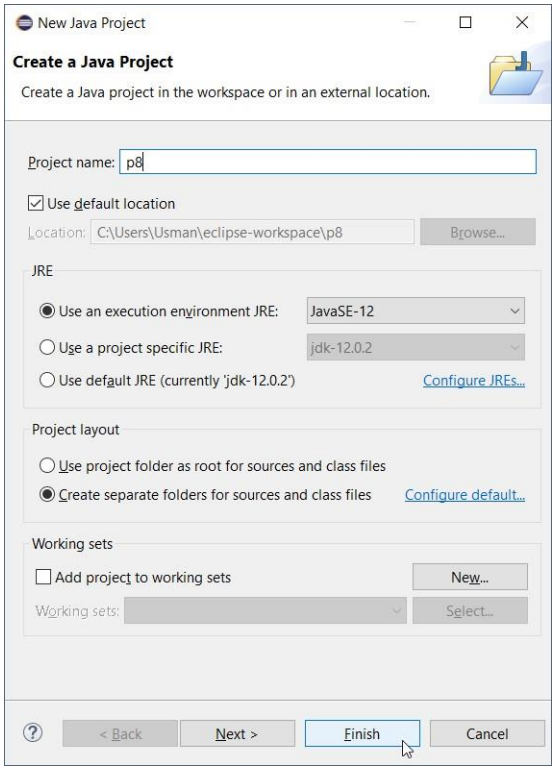
1) Open Eclipse. Select your workspace directory. Click Launch.



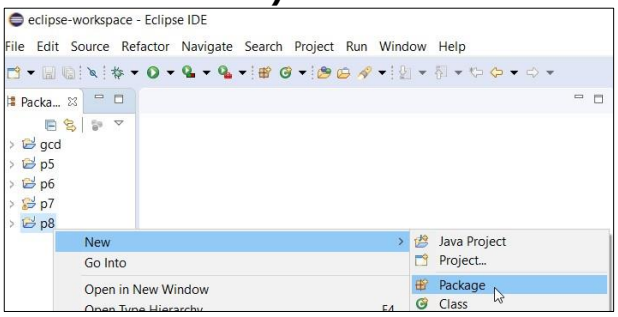
2) Create a Project (File > New > Java Project)

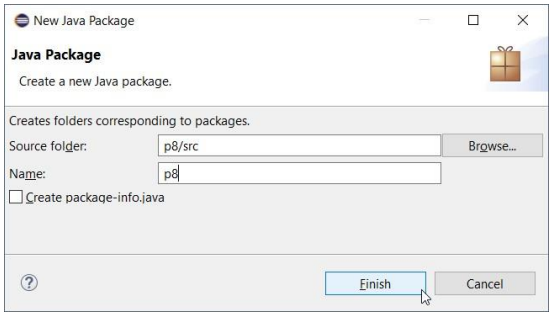


3) Name the project as “p7” > click Finish.

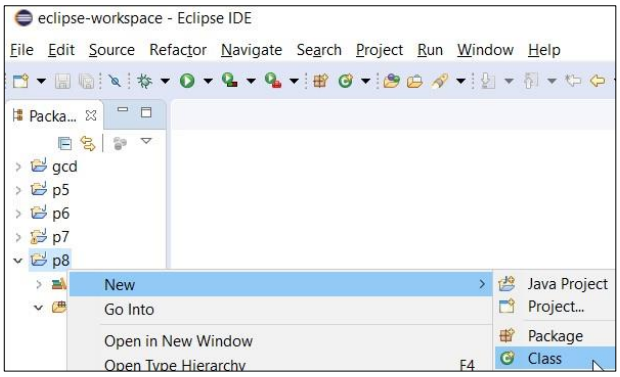


4) Create a Package(right-click on Project Name > New > Package > Name it > Finish)

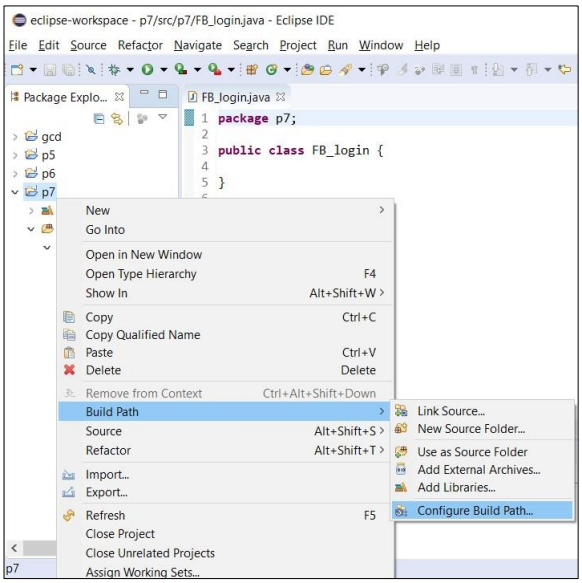




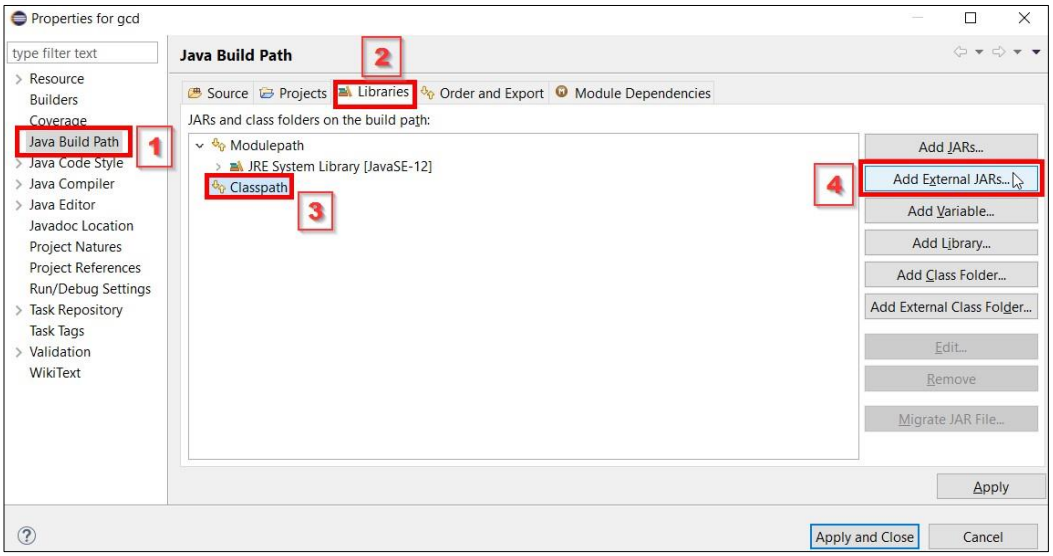
5) Create a Class(right-click on Project Name > New > Class > Name it > Finish)



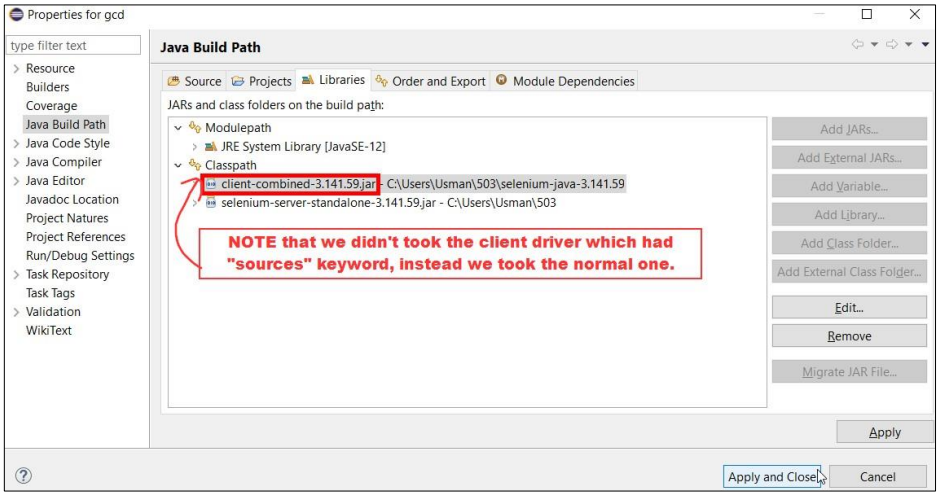
6) Adding “Selenium *Server Driver* and *Client Driver*(JAR files)” in Eclipse IDE: • right-click on Project Name > Build Path > Configure Build Path…



- **now go under: Java Build Path > Libraries > Classpath > click Add External JARs…**



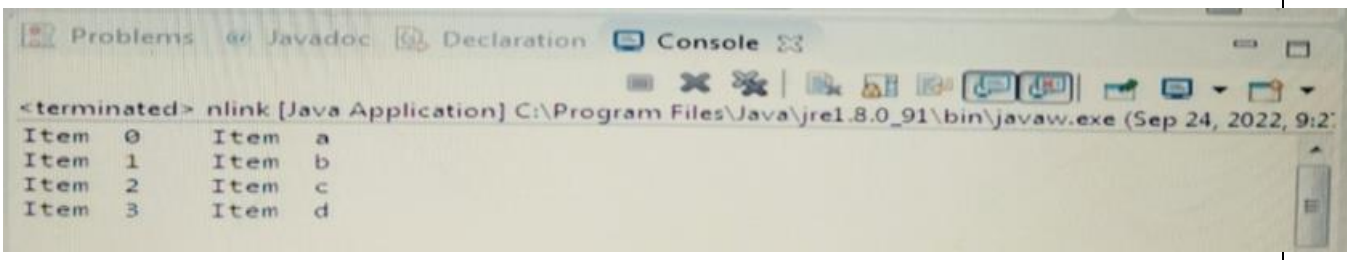
- **Browse and add JAR files > click Apply and Close :**




```
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-
v0.21.0win32\\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> items =
driver.findElements(By.tagName("option"));
```

```
 System.out.println("Total items are"+items.size());
 for (int i = 0; i<items.size(); i=i+1)
       {
System.out.println("Item  "+ i + "    Item name   "+ items.get(i).getText());
       }
}
}
```

Output:-



Practical No :- 8

Aim :- Write and test a program :-

a) to get the number of items in a list / combo box &

b) to count the number of check boxes on the page checked and unchecked count.

Steps:

(Follow the same procerdure that were used to perform practical no. 7 till adding external jars)

1) **Create HTML file in a notepad > Save it > Open it in browser after writing the code that file > Copy the URL and paste the copied URL**

a) **HTML Code for list:-**

```
<html>
<body>
<ol>
<li>  a  </li>
<li>  b  </li>
<li>  c  </li>
<li>  d  </li>
</ol>
</body>
</html>
```

b) **HTML Code for combobox:-**

```
<html>

<body> <select>

<option>Volvo</option>

<option>Express</option>

<option>Mercedes</option>

 <option>Raja hamsa</option>
```

```
</select

</body>

</html>
```

c) HTML Code for checkbox:-

```
<html>

<body>

<input type="checkbox" value="btn"/> <label> one </label>
<input type="checkbox" value="btn"/> <label> two </label>

<input type="checkbox" value="btn"/> <label> three</label>
<input type="checkbox" value="btn"/> <label>four</label>


</body>

</html>
```

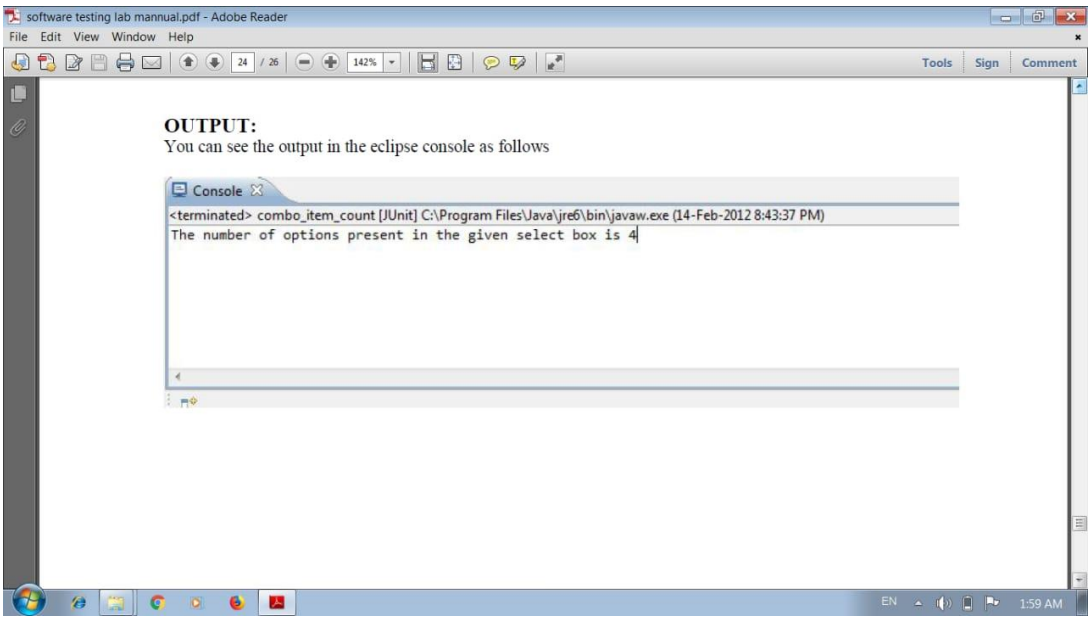
d) HTML Code for radiobutton

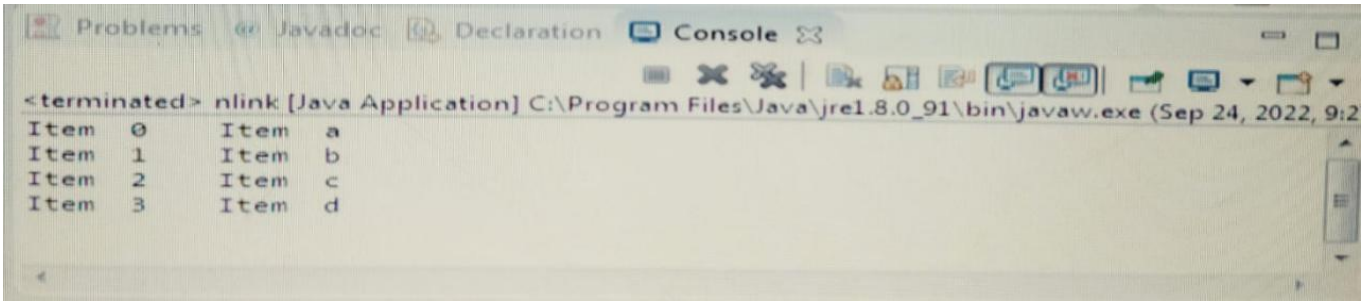
```
<html>

<body>

<input type="radio" value="btn"/> <label> one </label>
<input type="radio" value="btn"/> <label> two </label>

<input type="radio" value="btn"/> <label> three</label>

<input type="checkbox" value="btn"/> <label>four</label>

</body>
```

</html>

Output:-

(Output for list)





(Output for combobox)