Table of Contents

1. [Exercise 2.1: Add PyDev Plugin in Eclipse 3](#_Toc507520404)
2. [Exercise 2.2: Defining Objects and Functions in Python 4](#_Toc507520405)
3. [Exercise 2.3: Illustrate Exception Handling in Python 5](#_Toc507520406)
4. [Exercise 3.1: Cross Browser Testing 7](#_Toc507520407)
5. [Exercise 4.1: Identifying Web Elements 9](#_Toc507520408)
6. [Exercise 4.2: Using Select Command 11](#_Toc507520409)
7. [Exercise 5.1: Test Execution for New Tours demo 13](#_Toc507520410)
8. [Exercise 5.2: Executing Test as Test Suite 15](#_Toc507520411)
9. [Exercise 6.1: Read Data from Excel using openpyxl library 17](#_Toc507520412)
10. [Exercise 7.1: Implementing Page Object Model design pattern 19](#_Toc507520413)
11. [Exercise 8.1: Retrieve the address of a city in About Us for TestMeApp 22](#_Toc507520414)

Module 2

# Exercise 2.1: Add PyDev Plugin in Eclipse

## Prerequisite

Participants should have a prior understanding of basics of Python

## Steps

|  |  |
| --- | --- |
| 1 | Open Eclipse -> Go to Help -> Market place |
| 2 | In Market Place - > Search box -> search for word “ Pydev” |
| 3 | In Search result - > check the box for Pydev for Eclipse - > confirm |
| 4 | System will prompt for Eclipse Restart -> Click Restart now |
| 5 | Pydev Plugin is now installed in the Eclipse |

# Exercise 2.2: Defining Objects and Functions in Python

## Scenario

|  |
| --- |
| Create a simple calculator that can Add, Subtract, Multiply and divide two numbers depending upon the input from the user |

## Prerequisite

Participants should have a prior understanding of basics in Python.

## Steps

|  |  |
| --- | --- |
| 1 | Open the new notepad file |
| 2 | Save file as **Solution3.2.py** (Python file extenstion is .py) |
| 3 | Enter the below snippet : |
| 5 | Save the file |
| 6 | Open the command prompt and in the command prompt open the respective folder which contains Solution3.2.py file. |
| 7 | Write command **<file\_name.py> i.e. Solution3.2.py** to execute the source code and click **enter** |

# Exercise 2.3: Illustrate Exception Handling in Python

## Scenario

|  |
| --- |
| Write a program to handle divide by zero exception |

## Prerequisite

Participants should have a prior understanding of Iterative constructs and basics in Python.

## Steps

|  |  |
| --- | --- |
| 1 | Open Python |
| 2 | Enter the below snippet:  >>> 10/0  Traceback (most recent call last):  File "<stdin>", line 1, in <module>  ZeroDivisionError: division by zero  **Note : ZeroDivisionError** |
| 3 | Open a new test in Python |
| 4 | Enter the below snippet :  Num1=input("Enter a Number1:")  Num2=input("Enter a Number2:")  try:  val=int(Num1) / int(Num2)  except ZeroDivisionError as e:  print ('ZeroDivisionError Exception') |
| 5 | Run the test |
| 6 | Output: |

# Exercise 3.1: Cross Browser Testing

## Scenario

|  |
| --- |
| Design a test script in python using selenium python API and launch the new tours demo in different browsers like IE, Chrome , Firefox. |

## Prerequisite

* + - Selenium requires a driver to interface with the chosen browser.
    - Download drivers for different browsers from <http://www.seleniumhq.org/download>
    - Set the PATH in the Environment Variables.  
      For example Google Chrome requires chromedriver.exe, which needs to be downloaded & Set.
    - For Internet Explorer, Protected mode must be enabled
      * Open Internet Explorer
      * Choose Tools-> Internet options -> Security
      * Enable Protected mode under Local Internet & Trusted sites
    - Also set browser zoom level to 100% in Internet Explorer

## Steps

|  |  |
| --- | --- |
| 1 | Create a pydev project and open a pydev module |
| 2 | Type the below code in the python file  *from selenium import webdriver*  *browser = webdriver.Chrome(“path to chromedriver.exe”)*  *browser.get('http://newtours.demoaut.com/’)*  *browser1 = webdriver.Ie(“path to IEDriverServer.exe”)*  *browser1.get('http://newtours.demoaut.com/’)*  *browser2 = webdriver.Firefox(“path to geckodriver.exe”)*  *browser2.get('http://newtours.demoaut.com/')* |
| 3 | Execute the python as python run |
| 4 | **Output :**  Launch all <http://newtours.demoaut.com/> in browser like Chrome, IE , Firefox |

# Exercise 4.1: Identifying Web Elements

## Scenario

|  |
| --- |
| Locate the web elements in ‘TestMeApp’ for Sign-In page |

## Prerequisite

Open TestMeApp <http://localhost:8083/TestMeApp/> on a web browser. Example: Chrome

## Steps

|  |  |
| --- | --- |
| 1 | Create a pydev project and open a pydev module |
| 2 | Open the Sign in Page and note down the various WebElements like username, password and sign in button |
| 3 | Design a test script to login in the TestMeApp   * Open TestMe app * Login to the site using valid credentials * Print the username of the Logged In user on console * Logout * Close browser |
| 4 | Run the Python module  **Output:** |

# Exercise 4.2: Using Select Command

## Scenario

|  |
| --- |
| Use Select Command to select Paris city in <http://newtours.demoaut.com/> |

## Prerequisite

Open new tours website <http://newtours.demoaut.com/>

## Steps

|  |  |
| --- | --- |
| 1 | Create a pydev project and open a pydev module |
| 2 | To Do   * Open newtours website <http://newtours.demoaut.com/> * Login using valid credentials * Choose number of passengers as 3 * Choose Departing city as Paris * Choose a date from next month * Choose Arriving city as London * Click on Continue * Logout * Close browser |
| 3 |  |
| 4 | Run the Python module  **Output Screen 1:** |
| 5 | **Output Screen 2:** |

# Exercise 5.1: Test Execution for New Tours demo

## Scenario

|  |
| --- |
| Create a test class with a new test function to check the login functionality along with setUp () and tearDown() functions in the same test class |

## Prerequisite

Open TestMeApp <http://newtours.demoaut.com/> on a web browser like Chrome.

**To Do**

1. Create a setUp() life cycle function
2. Create a test function
3. Create a tearDown() life cycle function
4. Execute the Script as python unittest

## Steps

|  |  |
| --- | --- |
| 1 | Create a pydev project and open a pydev module |
| 2 | Create a setUp() function in the test class as shown below |
| 3 | Create a test function as shown below |
| 4 | Create a tearDown() life cycle function |
|  | Below shown is the complete code |
| 5 | Execute the script as python unittest  **Output:** |

# Exercise 5.2: Executing Test as Test Suite

## Scenario

|  |
| --- |
| Create a Test Suite class for testcases for newtours.demoaut.com and TestMeApp and execute the test scripts of the test suite |

## Prerequisite

Open TestMeApp <http://localhost:5432/TestMeApp/> on a web browser like Chrome.

**Note**: Using the TestSuites feature of unittest, we can collect various tests into logical groups and then into a unified test suite that can be run with a single command. This is done by using the TestSuite, TestLoader, and TestRunner classes.

To Do

1. Create Mercury Tours Test as Test Case
2. Create TestMeApp as Test Case
3. Create TestSuite class which can run all the Test Cases
4. Execute the runner

## Steps

|  |  |
| --- | --- |
| 1 | Create a pydev project and open a pydev module |
| 2 | Create Mercury Tours Test as Test Case |
| 3 | Create TestMeApp as Test Case |
| 4 | Create TestSuite class which can run all the Test Cases |
| 6 | Execute the runner  **Output:**  s |

# Exercise 6.1: Read Data from Excel using openpyxl library

## Scenario

|  |
| --- |
| Read the data from excel and send to TestMeApp login page with different sets of data. |

## Prerequisite

Open TestMeApp <http://localhost:8083/TestMeApp/> on a web browser like Chrome

**To Do**

1. Create an Excel sheet with data set
2. Create a test function
3. Create a separate login function
4. Create a seperate readExcel function
5. Run the script as python unit test

## Steps

|  |  |
| --- | --- |
| 1 | Create a pydev project and open a pydev module |
| 2 | **Create an Excel sheet with data set**    Save the excel sheet as data/Book1.xlsx under data folder in the pydev project |
| 3 | Create a test function |
| 4 | Create a separate login function as shown below |
| 5 | Create a seperate readExcel function |
| 6 | Run the script as python unit test  **Output:** |

# Exercise 7.1: Implementing Page Object Model design pattern

## Scenario

|  |
| --- |
| Implement Page Object Model as design pattern for TestMeAPP |

## Prerequisite

Open TestMeApp <http://localhost:5432/TestMeApp/> on a web browser like Chrome.

## Steps

|  |  |
| --- | --- |
| 1 | Create a pydev project and open a pydev module |
| 2 | Test the Login and Register page functionality of TestMeApp by implementing Page Object Model design pattern   * Create a separate class as Login Page * Create a separate class as Register Page * Create a Separate class for pefoming unitest on TestMeApp as shown below |
| 3 | Create a separate class as Login Page and capture all the object of the Registerpage as shown below |
| 4 | Create a separate class as Register Page and capture all the object of the Registerpage as shown below |
| 5 | Create a Separate class for pefoming unitest on TestMeApp as shown below |
| 6 | Run the test script as python unit test  **Output:** |

# Exercise 8.1: Retrieve the address of a city in About Us for TestMeApp

## Scenario

|  |
| --- |
| Retrieve the address of a city (Chennai / Bangalore) in About Us Tab for the TestMeApp by switching to windows / frames |

## Prerequisite

Open TestMeApp <http://localhost:5432/TestMeApp/> on a web browser like Chrome.

**To Do**

Create test to do the following actions

* Move to About US
* Move to Our Offices
* Move to Bangalore and click

Retrieve the office address of Bangalore city as text and display on console

## Steps

|  |  |
| --- | --- |
| 1 | Create a pydev project and open a pydev module |
| 2 | Below Shown Code to move to **About Us** . In the similar way move to **Our Offices** , move to **Bangalore** and click on **Bangalore**. |
| 3 | To Handle a new Window / Tab |
| 4 | To Handle frames |
| 5 | Run the test script as python unit test  **Output:** |