****

**Software Quality Assurance**

**Name:**

**Hamza Imran**

**Roll #:**

**22-10305**

**Section:** A

**Course:** Software Quality Assurance

**Instructor:** Dr. Saad Bin Saleem

**Report**

**What is unit test?**

JUnit inspired the unit test unit testing framework, which has a similar taste to prominent unit testing frameworks in other languages. It allows for test automation, the sharing of test setup and shutdown code, the grouping of tests into collections, and test independence from the reporting system.

To accomplish this, unittest uses an object-oriented approach to support a few key concepts:

**Test fixture:**

A test fixture is a representation of the steps required to conduct one or more tests, as well as any cleanup procedures. This could entail building temporary or proxy databases, directories, or initiating a server process, for example.

**test case:**

The specific unit of testing is known as a test case. It looks for a particular response to a specific set of inputs. TestCase is a foundation class provided by unittest that may be used to generate new test cases.

**test suite:**

A test suite is either a collection of test cases or a group of test suites. It's used to group tests that should all be run at the same time.

**test runner:**

A test runner is a component that orchestrates test execution and presents the results to the user. To indicate the results of running the tests, the runner can use a graphical interface, a textual interface, or return a particular value.

* **For This assignment I will be performing testing using PyUnit testing framework and wrote 6 test cases.**

**Methods:**

Several assert methods are provided by the TestCase class to check for and report failures. Following are the methods that I had used in this assignment:

| **Method** | **Checks that** |
| --- | --- |
| assertEqual(a, b) | a == b |
| assertNotEqual(a, b) | a != b |
| assertTrue(x) | bool(x) is True |
| assertFalse(x) | bool(x) is False |
| assertGreater(a,b) | a > b |
| assertLess(a,b) | a < b |

**Code Snippet of Functions on which testing will be applied:**

I have written down two functions:

* Function to calculate power (assertEqual(a, b), assertNotEqual(a, b), assertTrue(x), assertFalse(x) will be applied on this function).
* Function to determine whether a string is palindrome or not (assertGreater(a,b), assertLess(a,b) will be applied on this function).
* For the function of calculating power, I am taking the example of square of 2 i.e., **22 =4.**
* For the function of palindrome, I am taking string **“abba”** as palindrome and **“abca”** as not palindrome.

In these two functions I will perform PyUnit testing.

**Code:**

**Text

Description automatically generated**

**Test Cases:**

Following are the Test Cases that I had written for PyUnit testing:

**test\_module\_equal:**

**Text

Description automatically generated**

**test\_module\_notequal:**

**Text

Description automatically generated**

**test\_module\_greater:**

**Text

Description automatically generated**

**test\_module\_less:**

Text

Description automatically generated

**test\_module\_true:**

**A screen shot of a computer

Description automatically generated with low confidence**

**test\_module\_false:**

**Text

Description automatically generated**

**Test Suite for the above test cases is:**

**Text

Description automatically generated**

**Success Output:**

If we run the Test Suite containing all the test cases on the command line interface given that there is no issue with the code of both of the functions and with the test cases then we’ll get the result in such as way that all the 6 test cases will run successfully:

Text

Description automatically generated

**Individual Test Cases Success Output:**

**test\_module\_equal:**

**Text

Description automatically generated**

**test\_module\_notequal:**

Text

Description automatically generated

**test\_module\_greater:**

Text

Description automatically generated

**test\_module\_less:**

Text

Description automatically generated

**test\_module\_true:**

Text

Description automatically generated

**test\_module\_false:**

Text

Description automatically generated

**Failure Output Test Suite:**

Now we will check for failure cases by changing the values in the test cases. The result comes out to be:

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated