UFT Assignment Questions

1. Introduction to UFT:

- **Q1:** What is UFT (Unified Functional Testing)? How is it different from other test automation tools like Selenium or QTP?
- **Q2:** List the key features of UFT. Explain how it supports functional, regression, and GUI testing.
- **Q3:** What are the different types of objects that UFT can recognize? Give examples of each type.

2. Creating and Running a Basic Test in UFT:

- **Q4:** Create a simple test in UFT to open the Notepad application, type a text message, and save the file. Include the steps to record and run the test.
- **Q5:** Write a simple UFT script to open a web browser, navigate to a website (e.g., www.google.com), and perform a Google search.

3. Object Repository and Object Identification:

- **Q6:** What is an object repository in UFT? Explain the difference between "Local Object Repository" and "Shared Object Repository."
- **Q7:** Explain the concept of "Object Identification" in UFT. How does UFT recognize objects on the application being tested?

4. Checkpoints and Verification:

- **Q10:** What are checkpoints in UFT? Write a script to add a "Text Checkpoint" to verify that a specific text appears on a web page.
- **Q11:** Explain the difference between "Standard Checkpoints" and "Database Checkpoints" in UFT. Give an example of when you would use each.
- **Q12:** How can you handle dynamic objects using UFT? Explain with an example of handling dynamic buttons that change text based on user interactions.

5. Parameterization:

- **Q13:** What is parameterization in UFT? Why is it important for automating tests? Demonstrate how to parameterize a test using input data (e.g., user credentials for a login page).
- **Q14:** Create a test that accepts input parameters (e.g., username and password) from an Excel file and performs a login using that data.
- **Q15:** What are the different types of parameters available in UFT (e.g., test, action, and data table parameters)? Explain their use with examples.

6. Actions and Function Libraries:

- **Q16:** What is an action in UFT? How does it help in organizing your test scripts? Create an example of a reusable action for logging into a web application.
- **Q17:** Explain the concept of "Function Libraries" in UFT. How do you create and associate a function library with your test?
- **Q18:** Write a simple function in a UFT function library that accepts two numbers as inputs and returns their sum. Call this function from your test script.

7. Descriptive Programming:

- **Q19:** What is Descriptive Programming in UFT, and when would you use it? Write a UFT script using descriptive programming to click a button on a webpage (e.g., a "Submit" button).
- **Q20:** Explain the syntax for Descriptive Programming in UFT. Write a script that uses descriptive programming to interact with a web element based on its properties (e.g., link text, tagname, etc.).
- **Q21:** How does UFT handle dynamic objects with Descriptive Programming? Provide an example using a dynamic link or button.

8. Synchronization and Wait Statements:

- **Q22:** Why is synchronization important in UFT? What are the different synchronization techniques you can use to make sure your script waits for an element to be available?
- **Q23:** Write a script that uses the `Sync` method and `Wait` method to ensure UFT waits for a page to load before performing actions like clicking a button.
- **Q24:** How would you handle synchronization issues when testing a slow application or a page with dynamic content?

9. Error Handling and Recovery:

- **Q25:** How can you add exception handling in UFT to handle pop-ups or alerts that appear unexpectedly during the test execution?

10. Test Results and Reporting:

- **Q26:** Explain how UFT generates test results. How do you view and analyze the test results after running a test in UFT?
- **Q27:** What is the difference between the "Test Results" tab and the "Run-Time Data Table" in UFT? How would you use them to debug a failing test?
- **Q28:** Write a script that generates a custom report in UFT after executing a test case. This report should include test steps, status (pass/fail), and any relevant messages.

Answers

Q1:

UFT is a test automation tool for functional, regression, and GUI testing across web and desktop apps. Unlike Selenium, UFT supports desktop app testing, and it integrates API testing, unlike its predecessor QTP.

Q2:

Major features include object recognition, cross-platform support, data-driven testing, and integration with ALM. Functional testing validates requirements, regression ensures stability after changes, and GUI testing automates user interface interactions.

Q3:

UFT supports standard objects (e.g., `Button`), custom objects (e.g., charts), Windows objects (e.g., `WinButton`), and web objects (e.g., `WebEdit`). The above types facilitate testing of multiple applications.

Q4:

1. Open UFT and design a new test.

2. Start recording, open Notepad, type text, and save the file.

3. Stop recording and run the test to replay the actions.

Q6:

An object repository holds properties of objects identified by UFT. A **Local Object Repository** is specific to a test, while a **Shared Object Repository** can be reused across multiple tests.

Q7:

UFT uses properties like name, ID, or class to recognize objects. If primary properties fail, it uses assistive properties or ordinal identifiers for identification.

Q11:

Standard checkpoints verify properties of objects such as text while database checkpoints validate the content of the database like query results. Use standard for UI validation and database for backend validation.

Q12:

Dynamic objects are handled using regular expressions or descriptive programming. For example, a dynamic button can be identified as "innertext:=Submit.*".

Q13:

Parameterization runs tests with varying data to enhance coverage and reusability. For example, login tests can use multiple username-password pairs.

Q14:

Import the Excel file to the Data Table and parameterize the login test using `DataTable("Username")` and `DataTable("Password")`.

Q15:

- Test Parameters: Passed at runtime.

- Action Parameters: Share data between actions.

- Data Table Parameters: Use data from the Data Table (for example, login credentials).

Q16:

Actions help you break down the test scripts, hence increasing their reusability. For example, there is a login action that could be used as and when required, by different tests.

Q17:

Function libraries store reusable code. Create a `.vbs` file, write functions, and associate it with the test via "Settings > Resources."

Q19:

Descriptive Programming is where a script defines object properties directly instead of the object repository. It's useful for dynamic objects not present in the repository.

Q21:

UFT uses regular expressions or dynamic properties to locate objects such as the button with variable labels, represented as `\"innertext:=Submit.*\"`.

Q22:

Synchronization ensures UFT waits for the application to load before performing actions. Techniques include Sync, WaitProperty, and implicit wait settings.

Q24:

To resolve specific property waits or increasing the synchronization time globally via the UFT settings.

Q25:

Utilize Recovery Scenarios via UFT : Use triggers or recovery scenarios including things like: adding actions-define the same pop-up AND attaching it back to a Test using "Recovery Scenario Manager.

Q26:

UFT produces rich test results following a run, which are available from the "Results Viewer." It includes step-by-step status, screenshots, and error messages.

Q27:

The "Test Results" tab will display the details of execution and the "Run-Time Data Table" will provide the data during the execution of the test. Use both of them to debug the issues.