TISTA Science and Technology Corporation – Automation Engineer Mid / Sr.

Prescreen Assessment

Expected Completion time of 30-60 minutes depending on experience.

(If you don’t know how to solve any of the problems, feel free to leave them empty.)

Section A

1. How do you handle switching between multiple windows in Selenium WebDriver?

There is method in selenium WebDriver “switchto” use for switch the focus from one window to other as given by the below command

driver.SwitchTo().window(String windowHandle);

You can use the methods driver.getWindowHandles() to get all the open wondows ID and use the iterator to move to the next window.

When we need to switch the focus on the parent window, use the below command to switch back the focus on the parent window

driver.switchTo().defaultContent();

If you would like to move the focus on the other tab use the for loop to travers through the tab

**for** (String handle1 : driver.getWindowHandles()) {

System.***out***.println(handle1);

driver.switchTo().window(handle1);

}

The window can be close by using the method driver.close() and can be quit all window by using the the driver.quit method.

1. How do you handle pop-ups using Selenium WebDriver?

When we get the pop up window, we need to switch the focus on the popup window using the switchto method of the webdriver interface as given

driver.SwitchTo().window(String windowHandle);

Once you perform the action on popup then you need to switch back the focus on the parent window

driver.SwitchTo().window(Mainwindow);

1. Is there any way to verify the color change of the element using Selenium WebDriver? For example, there is the map of the US and when you select the state of Maryland, the Maryland will become red in the map (other parts are still gray).

We can use the getCssValue() method to get the color of the element and use the assert to verify the color. For example

String headerColor = driver.findElement(By.*xpath*("//\*[@id=\"cms\_Salesforce\_tileid\"]/i")).getCssValue("color");

System.***out***.println("headerColor ---->"+headerColor);

**try** {

Assert.*assertEquals*( headerColor,"rgba(255, 255, 255, 1)" );

System.***out***.println("color is matched");

}**catch**(Throwable t) {

System.***out***.println("color is not matched");

}

1. What is the different between Scenario and Scenario Outline in Cucumber?

Scenario is one of the core Gherkin structures. Every scenario starts with the keyword “Scenario:” and is followed by an optional scenario title. Each feature can have one or more scenarios and every scenario consists of one or more steps. For Example-

Scenario − Verify Help Functionality.

Given user navigates to Facebook.

When the user clicks on Help, then the Help page opens.

Scenario outline is similar to scenario structure; the only difference is the provision of multiple inputs. Scenario outline basically replaces variable/keywords with the value from the table. Each row in the table is considered to be a scenario.

the Scenario Outline keyword can be used to repeat the same steps with different values or arguments being passed to the step definitions.

Example

Scenario Outline − Login functionality for a social networking site. The given user navigates to Facebook.

When the user logs in using the Username as "<username>" and the Password as "<password>", then login should be successful.

| username | password |

| user1 | password1 |

| user2 | password2 |

1. Given these contents of the Customers table:

Id   Name                       ReferredBy

1    John Doe                 NULL

2    Jane Smith              NULL

3    Anne Jenkins         2

4    Eric Branford         NULL

5    Pat Richards           1

6    Alice Barnes           2

7    Jane Smith             3

8    Anne Jenkins         2

Here is a query written to return the list of customers not referred by Jane Smith:

      SELECT Name FROM Customers WHERE ReferredBy <> 2;

What will be the result of the query? Why? What would be a better way to write it?

Write a query that returns the unique names.

Result =

Pat Richards

Jane Smith

there are 5 customers not referred by Jane Smith (including Jane Smith herself), the query will only return two: Pat Richards and Jane Smith. All the customers who were referred by nobody at all (and therefore have NULL in their ReferredBy column) don’t show up. But certainly those customers weren’t referred by Jane Smith, and certainly NULL is not equal to 2, so why didn’t they show up?

SQL Server uses three-valued logic, which can be troublesome for programmers accustomed to the more satisfying two-valued logic (TRUE or FALSE) most programming languages use. In most languages, if you were presented with two predicates: ReferredBy = 2 and ReferredBy <> 2, you would expect one of them to be true and one of them to be false, given the same value of ReferredBy. In SQL Server, however, if ReferredBy is NULL, neither of them are true and neither of them are false. Anything compared to NULL evaluates to the third value in three-valued logic: UNKNOWN.

The query should be written:

SELECT Name FROM Customers WHERE ReferredBy IS NULL OR ReferredBy <> 2

    John Doe

    Jane Smith

    Eric Branford

    Pat Richards

   Jane Smith

SELECT DISTINCT Name FROM Customers WHERE ReferredBy IS NULL OR ReferredBy <> 2

    John Doe

    Jane Smith

    Eric Branford

    Pat Richards

Section B

1. Add a method which takes a string and returns a version without the first and last char. For example, "Hello" should return "ell". The string length will be at least 2.

public void removefirstandlaststring( String text) {

String withoutFirstandLastCharacter = text.substring(1, text.length() - 1);

System.out.println("input string: " + text);

System.out.println("without First and last character: " + withoutFirstandLastCharacter);

}

1. Add a method which takes a list of integers, return a list of the integers, omitting any that are less than 0.

Section C

1. Write a method or functions in the major programming language of your choice that returns the longest word in a sentence and its length. For example, “The quick brown fox jumped over the lazy dog.” should return “jumped” and 6.

**package** practice;

**public** **class** LongestWord

{

**public** **static** **void** main(String[] args)

{

String s="The quick brown fox jumped over the lazy dog.";

String[] word=s.split(" ");

String rts=" ";

**for**(**int** i=0;i<word.length;i++){

**if**(word[i].length()>=rts.length()){

rts=word[i];

}

}

System.***out***.println(rts);

System.***out***.println(rts.length());

}

}

1. Write unit tests, reworking code as needed.
2. Add a method that returns the shortest word and length with unit tests.
3. Create a README documenting any assumptions you made and including instructions on how to build and execute your tests.
4. Share your code and answers using GitHub or similar.