**Walmart Interview Questions:**

1. current project and its end to end architecture of automation.
2. [how a cron job is set up for running the automated tests](#_2._How_a)
3. how do we test the mobile application with appium.
4. [4 major components of selenium](#_4._Four_major).
5. what is the needs of automation when we are doing the integration testing.
6. how do we do integration testing
7. how bdd test scenario's work
8. how to set up the pipeline to run the tests.
9. how much time does it take to run the current project automated test scenarios.
10. is there a way we can reduce the time and run these test multiple times a day.
11. who owns github for automation code devvelooper or automation engineer.
12. different types of framework that i have worked on
13. how do automation engineer make use of DB.
14. no sql database use cases as an automation engineer.
15. [Difference between Overloading and Overriding](#_15._Difference_between)
16. Explain your current project Framework?
17. Explain Implicitly wait() & explicitly wait()
18. Difference between CSS and xpath?
19. How to access browser?
20. [Why WebDriver driver=new FirefoxDriver () why not WebDriver driver= new WebDriver()?](#_20._Why_WebDriver)
21. [What is POM?](#_21._What_is)
22. [Difference between POM and page factory](#_22._Difference_between). -
23. [Explain Error,bug and defect](#_23._Difference_between)
24. Tell me about TestNG
25. Basic Tools for Mobile Automation testing?
26. [Explain Hooks in cucumber](#_26._Explain_Hooks).
27. [What is the difference between Background and Scenario](#_27._What_is)
28. [How do you pass data value input to a scenario in feature file](#_28._How_do)
29. [Important parameters of Cucumber Runner class](#_28._Important_parameters)
30. [Java Program to print word count](#_30._Java_Program)
31. [Java program to print Alphabets A to Z](#_31._Java_program)
32. Java program for add 2 number to match the given target value and print the indexes.

**ANSWERS:**

# **2. How a cronjob is set up for running the automated tests?**

Ans.

A cronjob is set up using the following syntax:

\* \* \* \* \* command to execute

Here the first \* indicates minutes of the hour (0-59)

the second \* indicates the hour of the day (0-23)

the third \* indicates the day of the month (1-31)

the fourth \* indicates the month of the year (1-12)

and the fifth \* indicates the day of the week(Sun – Sat as 0-7 where both 0 and 7 indicate Sunday)

We create a cron table or crontab using the command crontab -e. For eg. If we want to run the maven test class “FirstTest” at midnight every Monday, the we use the following command:

1. 0 \* \* 1 mvn test -Dtest=FirstTest

[Top](#_top)

# **4. Four major components of selenium.**

Ans.

Test Framework Host System

WebDriver

Browsers

Selenium Server or Grid

Drivers

The above diagram shows all the components of selenium when the Host System is remotely connected to the WebDriver through the Selenium Server or Grid. When the WebDriver, Drivers and Browsers are part of the same Host System, it is called direct communication. In such a case, Selenium Server or Grid is not part of the system.

Remote communication can be in two ways. It can either be through a Remote WebDriver or through Selenium Server or Grid. If communication is through the Remote WebDriver, then we have a WebDriver component outside the host system and a Remote WebDriver inside the host system with the Drivers and Browsers. While using the Selenium Server, both the WebDriver and the Selenium Server are outside the host system.

The Selenium Grid is used mainly for running tests in parallel in multiple machines. The WebDriver is initiated from the test framework like TestNG or Cucumber and communicates with the server or Grid which in turn connects to the drivers that run the browsers.

[Top](#_top)

# **9. how much time does it take to run the current project automated test scenarios.**

Ans.

The current project has around 350 test cases which takes around 4 hours to run.

# **15. Difference between Overloading and Overriding**

Ans.

A method is said to be *overloading*, if there is another method with the same name but the input or output parameters of the method is different from each other.

For eg.

public int add(int a, int b);

public double add(double a, double b);

A method is said to be *overriding*, if there is another method with same name as well as same input and output parameters. This usually happens during inheritance when the child has the same function as the parent but different lines of code.

[Top](#_top)

# **20. Why WebDriver driver=new FirefoxDriver () why not WebDriver driver= new WebDriver()?**

Ans.

WebDriver is an interface and not a class. FirefoxDriver() is the class that has implemented the interface WebDriver. The interface is a complete abstract class that groups methods but the methods have no body. The body of the methods in an interface are provided by the class that implements it using the keyword ‘implements’.

[Top](#_top)

# **21. What is POM?**

Ans.

POM refers to Page Object Model. POM is a design pattern commonly used in Selenium for automating test cases. This pattern can be used with any Test Framework like Data Driven, Keyword Driven, Hybrid etc. Here web pages are represented by a corresponding class, the web elements are the variables of the class and the all interactions are provided by the methods of the class

[Top](#_top)

# **22. Difference between POM and page factory.**

Ans.

POM is a design pattern that organizes page objects into pages such that the pages and the test scripts can be easily differentiated. Whereas Page Factory is a class in Selenium that helps implement the Page Object Model.

In POM, the elements are located using the ‘By’ keyword whereas in Page Factory, elements are located using ‘@FindBy’ annotation.

POM does not handle exceptions well but Page Factory handles it well.

POM does not perform lazy initialization but Page Factory does.

POM uses a cache storage while performing tasks whereas Page Factory does not.

[Top](#_top)

# **23. Difference between Error, Defect and Bug**

Ans.

An error is a mistake made during coding. When a developer cannot successfully compile or run a program due to a coding mistake, it is called an error.

A defect is when the application does not work as expected. This is found by the programmer inside the code during development.

A bug is a defect that is found by the testing team that is reproduced and recorded.

[Top](#_top)

# **26. Explain Hooks in cucumber.**

Ans.

Cucumber uses Feature files as the starting point of their framework. Each feature file can have any number of Scenarios. It is possible that these scenarios can have some common lines or blocks of code which need to be implemented before or after each scenario. Such lines of code can be grouped together and can form hooks. There are four types of hooks. @Before, @After, @BeforeStep and @AfterStep.

@Before hook contains blocks of code that should be executed before each Scenario is executed. @After hook contains code that should be executed after each Scenario. @BeforeStep is executed before each step in all scenarios and @AfterStep is executed after each step.

Hooks can also be combined with tags such that those blocks of code will be executed only for those Scenarios that have been similarly tagged.

Hooks are written in a separate class file and is not visible in the feature file.

[Top](#_top)

# **27. What is the difference between Background and Scenario**

Ans.

Scenario is a list of one or more steps in the Feature file in the Cucumber framework that represents a test case.

Background is a list of one or more steps in a Feature file that is common to all the Scenarios and which is to be executed before each Scenario.

[Top](#_top)

# **28. How do you pass data value input to a scenario in feature file**

Ans.

There are three ways in which to add data value input into a Scenario in a Feature file.

The first way is to directly hard code the data into the step of the Scenario. Changes are made to the step definition file to accept the data.

The second way is to change the keyword Scenario to Scenario Outline. The steps in the Scenario Outline and followed by the keyword Examples where the data table is provided that contains two or more sets of data with column headers. These headers are mentioned in the Scenario Outline steps where data needs to be sent.

The third way is to retain the Scenario keyword and provide the data as a table after each step that accepts the data.

[Top](#_top)

# **29. Important parameters of Cucumber Runner class**

Ans.

There are several parameters in the Cucumber Runner class mentioned as @CucumberOptions. Out of them, two are the most important and needs to be present in the file. They are ‘features’ and ‘glue’. The ‘features’ options gives us the location of the cucumber feature files that need to be run. The ‘glue’ option specifies the location of the step definition files.

There are other parameters like ‘tags’. The ‘tags’ option specifies the tags of the scenarios that need to be run. If no ‘tags’ are specified, it will run all the scenarios of the feature files mentioned in the ‘features’ option.

[Top](#_top)

# **30. Java Program to print word count**

Ans.

public class WordCountInString {

public static void main(String[] args) {

int count = 0;

String sentence;

int strLength;

Scanner scan = new Scanner(System.***in***);

System.***out***.println("Enter the string for counting words:");

sentence = scan.nextLine();

scan.close();

strLength = sentence.length();

for (int i = 0; i < strLength; i++) {

if ((sentence.charAt(i) == ' ') && (sentence.charAt(i + 1) != ' ')) {

count++;

}

}

if (strLength > 1) {

count++;

}

System.***out***.println("The number of words in the string is: " + count);

}

}

[Top](#_top)

# **31. Java program to print Alphabets A to Z**

Ans.

public class PrintAtoZ {

public static void main(String[] args) {

// Prints a to z

for (int i = 97; i < 123; i++) {

System.*out*.println((char) (i));

}

// Prints A to Z

for (int i = 65; i < 91; i++) {

System.*out*.println((char) (i));

}

}

}

[Top](#_top)