Cucumber Interview QA

1. **What do you mean by Cucumber?**

Ans. Cucumber is a set of libraries used to achieve **Behaviour Driven Development.** BDD is a software development process for teams to create simple scenarios on how an application should behave from the end user's perspective.

2. **What is feature file?**

Ans. A feature file is one which contains scenarios on Application Under Test. Its first line should always start with keyword: 'Feature' which should describe which feature this file is testing.

3. **What are various keywords used in Feature file.**

Ans.

1. **Feature:**
   a. The purpose of the Feature keyword is to provide a high-level description of a software feature, and to group related scenarios.
   b. e.g Feature: Positive scenarios for Login Functionality
2. **Given, When, Then, And, But:** These keywords are used to write test steps. There is no rule defined for sequence of these keywords. We can use any keyword in any sequence. Even duplicate in one scenario. Just make sure to use these keywords in such a way that they should form a correct English statement.

   E.g. Given Browser is opened
   Then User will enter username and password
   And User will click on login button
   Then User should able to login to the application
3. **Scenario:** Each feature file will have multiple test Scenarios. These test cases are written under keyword 'Scenario'

   E.g. Scenario: Enter valid username and password
   Given Browser is opened
   Then User will enter username and password
   And User will click on login button
   Then User should able to login to the application

4. **Background:** If there are some common steps that we need to execute just before execution of every scenario, then we can add those steps under 'Background' keyword instead of 'Scenario' keyword. This is similar as @BeforeMethod of TestNG.

E.g. Background: User is logged in
Given Browser is opened
Then Navigate to login page url
Then Enter valid username and password
And Click on login button.

5. **Scenario Outline:** The Scenario Outline keyword can be used to run the same Scenario multiple times, with different combinations of input (Test Data).
   E.g. Scenario Outline: Login with different uname and pass values
   Given Browser is opened with login page url
   Then Enter <uname>
   And Enter **<**password**>**
   And Hit Sign in Button
   Examples:
uname	password
firstuser	first@123
seconduser	second@123
thirduser	third@123
   Using above code, we can perform Data Driven Testing
   Here one more keyword we have user, that is 'Examples'. This keyword is used to specify a set of data enclosed by pipes (|)
6. **#language:** This keyword is used to set the language of test scenarios. If we don't specify it, Cucumber will automatically consider language as English. If we set this keyword to (#language: hi) then we can write the scenarios in Hindi also.

4. What programming languages are supported by Cucumber?
Ans. Java, .Net, PHP, Python, Ruby etc.

5. What are the advantages of Cucumber Framework.
Ans.
- Cucumber is an open source tool.
- Plain Text representation makes it easier for non-technical users to understand the scenarios.
- It bridges the communication gap between various project stakeholders such as Business Analysts, Developers, and Quality Assurance personnel.
- Automation test cases developed using the Cucumber tool are easier to maintain and understand as well.

- Easy to integrate with other tools such as Selenium and Capybara.
- No need to maintain separate Test Documentation.

6. **Tell me list of jars required for Cucumber.**

   Ans.

   - cucumber-core.jar
   - cucumber-java.jar
   - cucumber-junit.jar
   - cucumber-jvm-deps.jar
   - cucumber-reporting.jar
   - gherkin.jar

   Make sure that, cucumber-core, cucumber-jvm and cucumber-junit have same version. Otherwise it may lead to some weird errors.

7. What is @CucumberOptions ?

   Ans. Cucumber Options tag is used to provide a link between the feature files and step definition files. Each step of the feature file is mapped to a corresponding method on the step definition file.

   E.g. `@CucumberOptions(features="Features",glue={"StepDefinition"})`

   Where, "Features" is path to folder in which feature files to execute are saved. And "stepDefinition" is package in which mapping step definitions are written. (Remember, Cucumber steps have to be mapped with Java methods or any language method that you are using with Cucumber)

8. **What are Hooks in Cucumber?**

   Ans. Hooks are simply annotations written for methods.

   There are two Hooks in Cucumber.

   a. @Before: Method with this annotation will execute before every scenario

   b. @After: Method with this annotation will execute after every Scenario

   E.g.

   Scenario: Testing Hooks first
   　　Given First Step
   　　Then Second Step
   Scenario: Testing Hooks second
   　　Given third step
   　　Then fourth step

```
import cucumber.api.java.After;
import cucumber.api.java.Before;

public class Hooks{
    @Before
    public void openChromeBrowser(){
        "Code to open chome browser"
    }
    public void closeChromeBrowser(){
        "Code to close chrome browser"
    }
}
```

9. **What is TestRunner class**
   Ans. Cucumber Scenarios can be executed in various ways like from command prompt. But the most preferred way is to run from TestRunner class. It is a simple Java class with two annotations. We should not write any code inside this class. We can provide any name to this class

```
import org.junit.runner.RunWith;
import cucumber.api.CucumberOptions;
import cucumber.api.junit.Cucumber;

@RunWith(Cucumber.class)
@CucumberOptions(features="Features",glue={"StepDefinition"})
public class Runner
{
}
```