**What Is Mockito?**

* Mockito is a Java-based framework used for unit testing of Java applications. This helps in the development of testable applications. You use Java Reflection API internally to generate mock objects. Mockito is used to simplify test development by mocking external dependencies and using them in the code.

The benefits or advantages of Mockito are explained in the following points:

* No handwriting - The developers do not need to write their Mock codes.
* Return values - Mockito supports the return values.
* Safe Refactoring - Even if an interface method is renamed or the parameters are reordered, the test codes created as Mocks will not break.
* Exception support - Mockito enables exceptions.
* Annotation support - Mockito supports the creation of Mocks with annotation.
* Order Support - Mockito allows checking on the order of method calls.

Mock Object

Mock is an Object that clone the behavior of a real object. It is basically used in Unit Testing by testing the isolated unit even when Backend is not available.

Stub()

It is an object which simulates real object with minimum number of method required to perform a test

Spy()

A Spy is like a partial mock, which will track the interactions with the object like a mock. Additionally, it allows us to call all the normal methods of the object. Whenever we call a method of the spy object, the real method will be invoked(unless it is stubbed).

test lifecycle

* A test class contains @BeforeAll, @BeforeEach, @Test, @AfterEach, @AfterAll annotations provided by Junit.
* The lifecycle methods executes before or after executing the test methods.
* @BeforeAll and @AfterAll annotations methods should be executed before all or after all the @Test test methods in the lifecycle.
* @BeforeEach and @AfterEach annotations methods should be executes before or after each test methods in the lifecycle.