1. What is JUnit?

Answer: JUnit is a unit testing framework for the Java Programming Language. It is written in

Java and is an Open Source Software maintained by the JUnit.org community.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2. What is a unit test case?

Answer: A unit test case is a small, isolated test that verifies the correctness of a specific piece of code,

typically a single function or method. It ensures that the individual units of a software application

function as intended, helping developers catch and fix errors early in the development process.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

3. What Is JunitTestCase?

Answer: JUnit is a testing framework for Java that provides a base for writing and running unit

tests. A JUnit test case is a Java class containing one or more methods annotated with @Test.

These methods represent individual test scenarios, and JUnit helps in running and managing

these test cases, providing assertions to verify expected outcomes.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

4. What are important features of JUnit?

Answer: Import features of JUnit are:

\* It is an open source framework.

\* Provides Annotation to identify the test methods.

\* Provides Assertions for testing expected results.

\* Provides Test runners for running tests.

\* JUnit tests can be run automatically and they check their own results and provide immediate feedback.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

5. What are JUnit classes? List some of them?

Answer: JUnit classes are important classes which are used in writing and testing JUnits.

Some of the important classes are:

Assert – A set of assert methods.

Test Case – It defines the fixture to run multiple tests.

Test Result – It collects the results of executing a test case.

Test Suite – It is a Composite of Tests.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

6. What are annotations and how are they useful in JUnit?

Answer: Annotations are like meta-tags that you can add to you code and apply them to

methods or in class. The annotation in JUnit gives us information about test methods ,

which methods are going to run before & after test methods, which methods run before

& after all the methods, which methods or class will be ignore during execution.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

7. What is a test suite?

Answer: A test suite is a collection or grouping of individual test cases that are executed

together. It allows developers to organize and run multiple tests as a single unit, making

it easier to manage and analyze the results of various test scenarios. Test suites are

commonly used in testing frameworks like JUnit to streamline the testing process.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

8. What is a junit fixture

Answer: In JUnit, a fixture refers to the preparation and setup needed for a test case to run.

It includes creating the necessary objects, initializing variables, and configuring the

environment before executing the actual test. The @Before annotation in JUnit is

commonly used to define fixture setup methods, ensuring consistency and repeatability

in test execution.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

9. What are the junit assert methods?

Answer: Some common JUnit assert methods include:

1. assertEquals(expected, actual): Checks if the expected and actual values are equal.

2. assertTrue(condition): Checks if the specified condition is true.

3. assertFalse(condition): Checks if the specified condition is false.

4. assertNull(object): Checks if the specified object is null.

5. assertNotNull(object): Checks if the specified object is not null.

6. assertSame(expected, actual): Checks if the expected and actual objects refer to the same object.

7. assertNotSame(expected, actual): Checks if the expected and actual objects do not refer to the same object.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

10. Parameterization ?

Answer: It allowing you to run the same test method with different data sets.

It involves running the same test logic with different input values. It is providing different

inputs to a test to validate software behavior under various conditions, improving test coverage.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

11. What is the difference between Mock or Stub?

Answer:

-- Stub

\* A dummy object that helps in running the test.

\* Provides fixed behavior under certain conditions which can be hard-coded.

\* Any other behavior of the stub is never tested.

For example, for an empty stack, you can create a stub that just returns true for empty()

method. So, this does not care whether there is an element in the stack or not.

-- Mock

\* A dummy object in which certain properties are set initially.

\* The behavior of this object depends on the set properties.

\* The object’s behavior can also be tested.