**1. What is Manual testing?**

Executing the test cases manually without any tool support is known as manual testing.

**2. How do you test a "private method" ?**

When a method is declared as "private", it can only be accessed within the same class. So there is no way to test a "private" method of a target class from any test class. Hence you need to perform unit testing manually. Or you have to change your method from "private" to "protected".

**3. What Are The Limitations Of Mockito?**

Cannot mock static methods

Cannot mock constructors

**4. How Should We Ignore Or Avoid Executing Set Of Tests ?**

We can remove @Test from the respective test so as to avoid its execution. Alternatively we can put @Ignore annotation on the Junit file if we want to ignore all tests in a particular file.

**5. Name Few Java Mocking Frameworks?**

Mockito, PowerMock, EasyMock, JMock, JMockit.

**6. Do You Mock Classes & Interfaces?**

Yes, the api is the same for mocking classes or interfaces.

**7. What is the purpose of org.junit.Assert class?**

This class provides a set of assertion methods useful for writing tests. Only failed assertions are recorded.

**8. Why not just use system.out.println () for Unit Testing?**

Debugging the code using system.out.println() will lead to manual scanning of the whole output every time the program is run to ensure the code is doing the expected operations. Moreover, in the long run, it takes lesser time to code Junit methods and test them on our files.

**9 .Why do you use Junit to test your code?**

Junit provides a framework to achieve all the following-:

Test early and often automated testing.

Junit tests can be integrated with the build so regression testing can be done at unit level.

Test Code reusage.

**10. Explain unit testing using Mock Objects.**

The common coding style for testing with mock objects is to −

* Create instances of mock objects.
* Set state and expectations in the mock objects.
* Invoke domain code with mock objects as parameters.
* Verify consistency in the mock objects.

**11. How To Test Whether The Returns Value Of The Method Is Expected ?**

Using Assert.

**12. How To Create A Junit To Make Sure That The Tested Method Throws An Exception ?**

@Test (expected = Exception.class)

**13. How Should We Ignore Or Avoid Executing Set Of Tests ?**

put @Ignore annotation on the Junit file if we want to ignore all tests in a particular file.

**14.Which class is a Composite of Tests?**

TestSuite

**15.Mention what are parameterized tests?**

Parameterized tests enable developer to perform the same test over and again using different values.

**16. How can we test methods individually which are not visible?**

using @VisibleForTesting annotation.

**17. With Junit 4, do we still need methods such as setUp and tearDown?**

No. This is taken care with help of @Before and @After annotations respectively

18**.What are called as test smells in relation with unit testing?**

Multiple assertions within one unit test, long-running unit tests etc

**19. Name some JUnit code coverage tools?**

Cobertura, EclEmma

**20. Which Of The Following Are Usually Automated And Which Are Executed Manually ?**

**1. Unit Test**

**2. Integration Test**

Unit Test are usually automated and Integration Tests are usually executed manually.

**21.Which method of TestSuite class returns the test at the given index?**

Test testAtintindex

**22.What is the purpose of org.junit.TestResult class?**

Answer:

A TestResult collects the results of executing a test case. It is an instance of the Collecting Parameter pattern. The test framework distinguishes between failures and errors. A failure is anticipated and checked for with assertions. Errors are unanticipated problems like an ArrayIndexOutOfBoundsException.

**23. Which method of TestResult class runs a TestCase?**

void run(TestCase test) method runs a TestCase.