**Core Java Questions**

**1.** Could you elaborate some Object Oriented Programming (OOP) Concepts and kindly describe each and every concept.

**2.** How would you implement method overloading and method overriding

**3.**  What are the 4 access modifiers in Java and kindly describe each and every access modifier.

**4.** What are the differences between an Abstract Class and an Interface

**5.** When a method in a class is declared as abstract, does not automatically makes the class abstract also?

**6.** Do all methods in an abstract class abstract in nature?

**7. What is the purpose of garbage collection in Java, and when is it used?**

**8.** What are Difference between Vector and Array List?

**9.** What are the Difference between Swing and Awt?

**10. What are the difference between a constructor and a method?**

**11. What is denoted by the final keyword in terms of class, methods and variables?**

**12.** What are the difference between Stringbuilder and Stringbuffer?

**13.** What is the difference between == and equals?

**14.** What is a transient variable?

**15**. Why do threads block on I/O?

**16.** How are Observer and Observable used?

**17.** Can a lock be acquired on a class?

**18.** Is null a keyword?

**19**. What is the Collections API?

**20.** How does Java handle integer overflows and underflows?

**21.** It uses those low order bytes of the result that can fit into the size of the type allowed by the operation.

**22.** What are wrapped classes?

**23.** Does garbage collection guarantee that a program will not run out of memory?

Garbage collection does not guarantee that a program will not run out of memory. It is possible for programs to use up memory resources faster than they are garbage collected. It is also possible for programs to create objects that are not subject to garbage collection.

**24.** Can an object's finalize() method be invoked while it is reachable?

An object's finalize() method cannot be invoked by the garbage collector while the object is still reachable. However, an object's finalize() method may be invoked by other objects.

**25.** Can a for statement loop indefinitely?

Yes, a for statement can loop indefinitely. For example, consider the following:

for(;;) ;

**26.** How many times may an object's finalize() method be invoked by the garbage collector?

An object's finalize() method may only be invoked once by the garbage collector.

**27.** Can a double value be cast to a byte?

Yes, a double value can be cast to a byte.

**28.** Does a class inherit the constructors of its superclass?

A class does not inherit constructors from any of its superclasses.

**29.** Can an abstract class be final?

An abstract class may not be declared as final.

**30.** What is the ResourceBundle class?

The ResourceBundle class is used to store locale-specific resources that can be loaded by a program to tailor the program's appearance to the particular locale in which it is being run.

**31.** How does get method of HashMap works in Java?

Yes, this is still one of the most popular questions for senior developer, you can expect this on telephonic round, followed by lot's of related questions, see here for answers

**32.** Which two method HashMap key object should implement?

(equals and hashcode)

**33.** Why should an object used as key should be Immutable?

(so that hashcode always return same value)

**34.** How does ConcurrentHashMap achieves it's scalability?

**35.** How do you share an object between threads?

there are multiple ways to do that e.g. Queues, Exchanger etc, but BlockingQueue using Producer Consumer pattern is the easiest way to pass an object from thread to another.

**36.** How do find if your program has deadlock?

( By taking thread dump using kill -3, using JConsole or VisualVM), I suggest to prepare this core java interview question in more detail, as Interviewer definitely likes to go with more detail e.g. they will press with questions like, have you really done that in your project or not?

**37.** How do you avoid deadlock while coding?

(By ensuring locks are acquire and released in an ordered manner)

**38**. What is busy spinning? Why should you use it?

One of the interesting multithreading question to senior Java programmers, Busy spinning is a waiting strategy, in which a thread just wait in a loop, without releasing CPU for going to sleep. This can be used in a particular scenario, where wait time is very minimal, by not releasing CPU or suspending thread, your thread retain all cached data and instruction, which may be lost if further suspended and resumed back in a different core of CPU. This question is quite popular in high frequency low latency programming domain, where programmers are trying for extremely low latency in range of Micro to Milli seconds.

**39**. What is ReadWrite Lock? Does ConcurrentHashMap uses ReadWrite Lock?

ReadWrite Lock is an implementation of lock stripping, where two separate locks are used for read and write operation. Since read operation doesn't modify state of object, it's safe to allow multiple access of shared object to multiple reader without locking, and by splitting lock into ReadLock and WriteLock, you can easily do that. Java provides an implementation of ReadWriteLock in form of ReentrantReadWritLock, which is worth looking. Also ConcurrentHashMap doesn't use ReadWriteLock, instead it divides maps into several segments and lock them separately using different locks, which means any given time, only a portion of map is locked, instead of whole map. This question is also very popular on Senior and experienced level Java interviews, expect Interviewer to go into more detail, e.g. asking you to provided an implementation of ReadWriteLock with different policies.

**40.** How to make an Object Immutable in Java? Why should you make an Object Immutable? Well, Immutability offers several advantage including thread-safety, ability to cache and result in more readable multithreading code. See here to learn how to make object Immutable. Once again, this question can also go into more detail and depending upon your answer, can bring several other questions e.g. when you mention Spring is Immutable, be ready with some reasons on Why String is Immutable in Java.

**41.** Which design patterns have you used?

Always expect design and patterns related question for Senior developer Core Java Interview. It's best to mention any GOF design pattern rather than Singleton or MVC, which almost every other Java developer use it. Your best bet can be Decorator pattern or may be Dependency Injection Pattern, which is quite popular in Spring Framework. It's also good to mention only design pattern, which you have really used in your project and knows it's tradeoffs. As once you mention a particular design pattern say Factory, Interviewer's next question would be, have you used in your project? So be ready with proper example and why you choose a particular pattern.

**42.** Do you know about Open Closed Design Principle or Liskov Substitution Principle?

Design patterns are based upon object oriented design principles. I strongly suggest to take a look at my article 10 SOLID and Object Oriented design principle, Java programmer should know, to at least have a basic idea of what are these principles and how they help you to write better object oriented code. If you don't know answer of this question, you can politely say no, as it's not expected from you to know answer of every question, but answering question, which most developer doesn't answer, can make your candidature much stronger.

**43.** Which design pattern you will use to shield your code from a third party library, which will likely to be replaced in another couple of years?

This is just another example of scenario based design pattern question, you can expect in different formats, some with more detail explanation with context, or some with only intent around. One way to shield your code form third party library is to depend upon interface rather than implementation and than use dependency injection to provide a particular implementation. This kind of questions are also asked quite frequently to experienced and senior Java developers with 5 to 7 years of experience.

**44.** How do you prevent SQL Injection in Java Code?

This question is more asked to Java EE developers than core Java developers but still a good question to know, PreparedStatement is the way to go. PreparedStatement not only provides better performance but also shield from SQL Injection attack. If you are working more on Java EE or J2EE side, than you should also be familiar with other security issues including Session Fixation attack or Cross Site Scripting attack and how to resolve them.

45. Tell me about different reference type available in Java, e.g. WeakReference, SoftReference or PhantomReference? and Why should you use them?

Well, they are different reference types coming from java.lang.ref package and provided to assist Java Garbage Collector in case of low memory issues. If you wrap an object with WeakReference than it will be garbage collected if there is no strong reference and GC is running low on memory. WeakHashMap is a Map implementation, whose keys uses WeakReference, so if only Map contains reference of any object and no other, those object can be garbage collected, if GC needs memory.

**JUNIT Questions**

**1.** What is testing?

Testing is the process of checking the functionality of the application whether it is working as per requirements.

**2.** What is unit testing?

Unit testing is the testing of single entity (class or method). Unit testing is very essential to every software company to give a quality product to their customers.

**3.** What is Manual testing?

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

**4.** What is Automated testing?

Taking tool support and executing the test cases by using automation tool is known as automation testing.

**5.** Out of Manual and Automated testing which one is better and why?

Manual Testing is:

-Time consuming and tedious.

-Huge investment in human resources.

-Less reliable.

-Non-programmable.

Automated testing is

-Fast

-Less investment in human resources

-More reliable

-Programmable

**6.** What is JUnit?

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.

**7.** What are important features of JUnit?

Important 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.

-JUnit tests can be organized into test suites containing test cases and even other test suites.

-JUnit shows test progress in a bar that is green if test is going fine and it turns red when a test fails.

**8.** What is a Unit Test Case?

A Unit Test Case is a part of code which ensures that the another part of code (method) works as expected. A formal written unit test case is characterized by a known input and by an expected output, which is worked out before the test is executed. The known input should test a precondition and the expected output should test a post condition.

**9.** When are Unit Tests written in Development Cycle?

Tests are written before the code during development in order to help coders write the best code.

**10.** Why not just use System.out.println() for 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 class files.

**11.** How to install JUnit?

**A:** Follow the steps below:

1. Download the latest version of JUnit, referred to below as junit.zip.

2. Unzip the junit.zip distribution file to a directory referred to as %JUNIT\_HOME%.

3. Add JUnit to the classpath:

set CLASSPATH=%CLASSPATH%;%JUNIT\_HOME%\junit.jar

4. Test the installation by running the sample tests distributed with JUnit (sample tests are located in the installation directory directly, not the junit.jar file). Then simply type:

java org.junit.runner.JUnitCore org.junit.tests.AllTests

All the tests should pass with an "OK" message. If the tests don't pass, verify that junit.jar is in the CLASSPATH.

**12.** Why does JUnit only report the first failure in a single test?

Reporting multiple failures in a single test is generally a sign that the test does too much and it is too big a unit test. JUnit is designed to work best with a number of small tests. It executes each test within a separate instance of the test class. It reports failure on each test.

**13.**In Java, assert is a keyword. Won't this conflict with JUnit's assert() method?

JUnit 3.7 deprecated assert() and replaced it with assertTrue(), which works exactly the same way. JUnit 4 is compatible with the assert keyword. If you run with the -ea JVM switch, assertions that fail will be reported by JUnit.

**14**.How do I test things that must be run in a J2EE container (e.g. servlets, EJBs)?

Refactoring J2EE components to delegate functionality to other objects that don't have to be run in a J2EE container will improve the design and testability of the software. *Cactus* is an open source JUnit extension that can be used for unit testing server-side java code.

**15.** What are JUnit classes? List some of them.

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.

TestCase - It defines the fixture to run multiple tests.

TestResult - It collects the results of executing a test case.

TestSuite - It is a Composite of Tests.

**16.** What are annotations and how are they useful in JUnit?

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.

**17.** How will you run JUnit from command window?

 Follow the steps below:

1. Set the CLASSPATH

2. Invoke the runner:

java org.junit.runner.JUnitCore

**18.** What is JUnitCore class?

JUnitCore is an inbuilt class in JUnit package. It is based on Facade design pattern. This class is used to run only specified test classes.

**19.**What is Test suite?

Test suite means bundle a few unit test cases and run it together. In JUnit, both @RunWith and @Suite annotation are used to run the suite test.

**20.** What is @Ignore annotation and how is this useful?

Sometimes it happens that our code is not ready and test case written to test that method/code will fail if run. The @Ignore annotation helps in this regards. Following are some of the usefulness of @Ignore annotation:

1. You can easily identify all @Ignore annotations in the source code, while unannotated or commented out tests are not so simple to find.

2. There are cases when you can't fix a code that is failing, but you still want to method to be around, precisely so that it does not get forgotten. In such cases @Ignore makes sense.

**21.** What are Parameterized tests?

 Parameterized tests allow developer to run the same test over and over again using different values.

**22.** How do you use test fixtures?

Fixtures is a fixed state of a set of objects used as a baseline for running tests. The purpose of a test fixture is to ensure that there is a well known and fixed environment in which tests are run so that results are repeatable. It includes:

1. setUp() method which runs before every test invocation.

2. tearDown() method which runs after every test method.

**23.** How to compile a JUnit Test Class?

Compiling a JUnit test class is like compiling any other Java classes. The only thing you need watch out is that the JUnit JAR file must be included in the classpath.

**24.** What happens if a JUnit Test Method is Declared as "private"?

 If a JUnit test method is declared as "private", it compiles successfully. But the execution will fail. This is because JUnit requires that all test methods must be declared as "public".

**25.** How do you test a "protected" method?

When a method is declared as "protected", it can only be accessed within the same package where the class is defined. Hence to test a "protected" method of a target class, define your test class in the same package as the target class.

**26.** 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".

**27.** What happens if a JUnit test method is declared to return "String"?

If a JUnit test method is declared to return "String", the compilation will pass ok. But the execution will fail. This is because JUnit requires that all test methods must be declared to return "void".

**28.** How can you use JUnit to test that the code throws desired exception?

JUnit provides a option of tracing the Exception handling of code. You can test if a code throws desired exception or not. The expected parameter is used along with @Test annotation as follows: @Test(expected)

**29.** What are Parameterized tests in JUnit?

Parameterized tests allow developer to run the same test over and over again using different values.

**30.** How to create Parameterized tests?

**A:** There are five steps, which you need to follow to create Parameterized tests:

1. Annotate test class with @RunWith(Parameterized.class).

2. Create a public static method annotated with @Parameters that returns a Collection of Objects (as Array) as test data set.

3. Create a public constructor that takes in what is equivalent to one "row" of test data.

4. Create an instance variable for each "column" of test data.

5. Create your tests case(s) using the instance variables as the source of the test data.

The test case will be invoked once per each row of data.

**31.** Can you use a main() Method for Unit Testing?

Yes you can test using main() method. One obvious advantage seems to be that you can whitebox test the class. That is, you can test the internals of it (private methods for example). You can't do that with unit-tests. But primarily the test framework tests the interface and the behavior from the user's perspective.

**32.** Do you need to write a test class for every class that needs to be tested**?**

No. We need not write an independent test class for every class that needs to be tested. If there is a small group of tests sharing a common test fixture, you may move those tests to a new test class.

**33.** When are tests garbage collected?

The test runner holds strong references to all Test instances for the duration of the test execution. This means that for a very long test run with many Test instances, none of the tests may be garbage collected until the end of the entire test run. Explicitly setting an object to *null* in the tearDown() method, for example, allows it to be garbage collected before the end of the entire test run.

**Enterprise Application Development/Miscellaneous Questions**

**1.** What Application Servers/Web Servers have you worked on?

**2.** What application frameworks/ORM tools have you worked on? Could you please provide the benefits of using that framework?

**3.** What front end technologies have you worked on?

**4.** What databases as well as querying tool have you used?

**5.** What is the extent of your knowledge in SQL? Are you comfortable in creating stored procedures/Triggers/SQL joins/Query Optimizations?

**6.** What File Versioning tool have you used?

**7.** What Project Management tool (i.e. Maven) that you used and its benefits?

**8.** What are the persistence technologies have you used?

**9.** Explain the concept of MVC and its uses.

**10.** Are you familiar with test driven development? Any testing or code analyzing tools/plugins that you used (i.e. JUNIT, JMETER, Sonarcube, PMD, Check style)

**11.** How can you establish a connection to the database using JDBC by means of a java class/using a Datasource?

**12.** What are Data Access Objects and how are they implemented?

**13.** What are the stages in the life cycle of servlet?

**14.** What are the differences between GET and POST?

**15.** How do you get the value of an input form in JSP?

**16.** Site some differences between a session and transaction.