1. What are the key differences between C++ and Java?

Answer: C++ and Java both are object-oriented programming languages with some differences. The interviewer may ask the difference between the two and include this in the top Java interview questions for freshers to check your basic knowledge. The differences between C++ and Java are as follows –

C++	Java
1) C++ is platform dependent.	1) Java is platform-independent.
2) C++ writes structural programs without using classes and objects.	2) Java is a pure object-oriented language except for the primitive variables.
3) C++ doesn't support documentation comments.	3) Java supports documentation comment (/**...*/) to create documentation for java code.
4) C++ fully supports pointers.	4) In Java, there is no concept of pointers.
5) C++ supports multiple inheritance.	5) Java doesn't support multiple inheritance.

2. Explain the JVM architecture?

Answer: Java Virtual Machine is the abstract machine or specification that provides a runtime environment to execute the bytecode. JVM supports Java and many other languages known as JVM languages, the program written in these languages is compiled into the bytecode and then executed by the JVM. contains key components which are classloader, memory area, execution engine etc.

a) Classloader

It is a subsystem of JVM which load class files. Whenever a Java program is run, it is loaded by the classloader.

b) Class Area

Class Area holds class-level data of each class file such as metadata, constant run pool, and static variables.

c) Heap

It is the runtime data which is used for allocating objects.

d) Stack

The stack is used for storing temporary variable. This component has a stack frame which is allocated one frame to each thread and when the execution of the thread is completed then that frame is also gets destroyed.

e) Registers

This component contains the address of JVM instruction which currently being executed.

f) Native Method Stack

All the native method stack used in the application are stored in this.

g) Execution Engine

It contains:

- A virtual processor
- An interpreter that executes the instructions after reading the bytecode.
- JIT compiler, used for improving the performance due to the slow execution. It compiles the similar part of the bytecode at the same time which reduces the total time needed for compilation.

h) Java Native Interface

It provides an interface which is needed for communicating with another application developed in another language like C, C++, C# etc.

3. What is the use of Classloader in Java?

Answer: A Java program is made up of a different number of custom classes and pre-defined classes. When a program is executed, JVM is used to load all the content of that needed class and through the use of Classloader JVM, it finds that class.

There are three types of Classloaders:

- **System Class Loader**

It loads all the classes from the classpath.

- **Extension ClassLoader**

It loads all the classes from the extension directory.

- **Bootstrap Class Loader**

It loads all the pre-defined java classes.

4. Which class is a superclass of all classes?

Answer: Java.lang.The object is the root class for all the java classes and we don't need to extend it. Every other java classes fall back under the object. All the different non-primitive types including arrays are inherited directly or indirectly from this class.

5. What is the static keyword?

Answer: The static keyword is used with a class level variable to make it global so all the objects will be able to share the same variable. It can also be used with methods. A static method can access only static variables of the class and invoke only a static method of the class.

The interview generally asks this question in the Java interview questions for freshers. Even if you are a fresher, you should have a good knowledge of the keywords in Java.

6. What are finally and finalize in Java?

Answer: Finally block is used with a try-catch block to put the code that you always want to get executed even the execution is thrown by the try-catch block. Finally is just used for releasing the resources which were created by the try block.

Finalize() is a special method in Object class that we can override in our classes. Finalize() is called by the Garbage collector to collect the garbage value when the object is getting it. This method is generally overridden to release the system resources when garbage value is collected from the object.

7. What is Type casting in Java?

Answer: Casting in Java is one of the top topics from where you can get questions in your interview. When we assign a value of one data type to a different data type then these two data types might not be compatible with each other and needs conversion. If data types are compatible with each other like, in case of the conversion of int value to long then

automatic conversion is done by Java and doesn't require typecasting. But if data types are not compatible with each other then they need to be cast for conversion.

Syntax

dataType variablename = (dataType) variableToConvert;

Also Read: Top 5 Java Frameworks

8. What is the inner and anonymous inner class?

Answer: In Java, we can define a class inside a class and they are called nested classes. Any nested class which is non-static are known as inner class. Inner classes are associated with objects of the class and they can access all the variables and methods of the outer class.

Any local inner class without any name is known as an anonymous inner class. It is defined and instantiated in a single statement. Anonymous inner class always extend a class or implement an interface. Since an anonymous inner class doesn't have any name, it is not possible to create its constructor.

9. What is break and continue statement?

Answer: In a while or do-while loop, we use break for a statement to terminate the loop. We use a break statement in a switch statement to exit the switch case. We can also use break statement for terminating the nested loop.

The continue statement is used for skipping the current iteration of a for, while or do-while loop. We can use the break statement with a label to skip the current iteration of the outermost loop.

The most basic programming question, not only related to the Java. If you have some knowledge of programming languages, you should know the answer to this question as it is among frequently asked Java interview questions for freshers.

10. What is an interface?

Answer: Interfaces are the core part of Java programming language used a lot in JDK, java design patterns, and most of the frameworks and tools. The interface provides a way to

achieve abstraction in Java and used to define the contract for the subclasses to implement.

The interface is a good starting point to define the type and create a top-level hierarchy in your code. In Java, a class can implement multiple interfaces, it's better to use interfaces as a superclass in most of the cases.

11. What is aggregation in Java?

Answer: Aggregation is best defined as the entity reference where it represents the relationship between two classes where the aggregate class contains a reference to the class which it owns. Aggregation represents a has-a and whole/part relationship.

For example consider an aggregate class Employee stores information such as name, age, salary, and the Address class stores information such as city, state, and pin-code. Now, if the Employee class is defined to contain an Address object then it will be said that the Employee object has-a Address object. The Address object also makes up part-of Employee object – there is no employee without any address to live. Therefore, the Employee object owns the Address object.

12. What is the use of System class in Java?

Answer: This question is among the most common Java interview questions for freshers. Java System class is one of the core classes. One of the easiest ways to log information for debugging is System.out.print() method. System class is final so we can't subclass and override its behavior through inheritance.

System class doesn't provide any public constructors, so we can't instantiate this class and that's why all of its methods are static. Some of the utility methods of System class are for array copy, get the current time, and reading environment variables.

13. What is an instanceof keyword?

Answer: We can use instanceof keyword in java to check whether an object belongs to a class or not. We should avoid much usage of it. Sample usage:

```
public static void main(String[] args) {



Object str = new String("abc");
```

```
If(str instanceof String) {

System.out.println("String value:" +str);

}

If(str instanceof Integer) {

  System.out.println("Integer value:" +str);

}

}
```

14. What is an Iterator?

Answer: Iterator interface provides methods to iterate over any collection. We can get iterator instance from a collection using iterator() method. Iterator takes the place of Enumeration in the Java Collection Framework. The iterator allows the caller to remove elements from the underlying collection during the iteration. Java Collection iterator provides a generic way for transversal elements of a collection and implements Iterator Design Pattern.

15. What is the Java Collections Framework?

Answer: Collections are used in every programming language and when initial java was released it contained few classes for collections: Vector, Stack, Array, and Hashtable. But

for larger scope and usage, Java 1.2 came up with Collection Framework that grouped all the collections interfaces, implementations, and algorithms.

Java Collection has come a long way with the usage of Generic and concurrent Collection classes for thread-safe operations. It has included blocking interfaces and their implementations in Java concurrent package.

16. What do you understand about Thread Priority?

Answer: Every thread when gets born is assigned with a priority value and usually higher priority gets precedence in execution but it also depends on the Thread Scheduler implementation which is OS dependent. We can assign the priority of thread but it doesn't guarantee that higher priority will get executed before lower priority thread. Thread priority is an integer value varies from 1 to 10 where 1 is the lowest and 10 is the highest priority thread.

17. What is Thread Scheduler and Time Slicing?

Answer: Thread Scheduler is Operating System service which allocates the CPU time to the available runnable threads. Once a thread is created and it's in the runnable phase then its execution depends on the implementation of the Thread Scheduler.

Time Slicing is a process of dividing available CPU time among the various runnable threads. Allocation of CPU time will depend on the thread priority or for how much time it is in the waiting state for getting the CPU time. Thread Scheduling cannot be controlled by Java, so it's always better to control it by the application itself.

18. Which is more preferred – Synchronized method or Synchronized block?

Answer: The synchronized block is more preferred because it doesn't lock the object, synchronized methods lock the object and if there are multiple synchronization blocks in the class, even though they are not related, it will stop the execution and put them in a wait state to get the lock on the object.

19. How to create daemon thread in Java?

Answer: Thread class setDaemon(true) is used for creating daemon thread in Java. We used to call this method before calling the start() method else it will give IllegalThreadStateException.

20. What is ThreadLocal?

Answer: ThreadLocal in Java is used for creating thread-local variables. We know that all threads of an object share its variables. So, if the variable is not threaded safe then we can use synchronization. But if we want to avoid synchronization then we can use ThreadLocal variables.

Every thread has its own ThreadLocal variable and they can use get() and set() methods to get the default value or change its local thread value. ThreadLocal instances are typically private static fields in classes that wish to associate the state with a thread.

21. Explain the Java Exception Hierarchy.

Answer: Java Exceptions are hierarchical and inheritance is used for categorizing the different types of exceptions. Throwable is the parent class of Java Exceptions Hierarchy and it has two child objects – Error and Exceptions.

Errors are exceptional scenarios which are out of the scope of applications and it's not possible to anticipate and recover from them, for example, hardware failure, JVM crash or out of memory error. Exceptions are further divided into checked and runtime exception.

Checked exceptions are exceptional scenarios that we can anticipate in a program and try to recover from it, for example, FileNotFoundException. We should catch this exception and provide a useful message to the user and log it properly for debugging purpose. The exception is the parent class of all the Checked exceptions.

Runtime exceptions are caused by bad programming, for example, trying to retrieve an element from the Array. At first, we should check the length of the array before trying to retrieve the element otherwise it might throw ArrayIndexOutOfBoundException at runtime. RuntimeException is the parent class of all runtime exceptions.

This is one of the most common Java interview questions for freshers. So, don't miss and clear your concepts on Java exceptions.

22. What happens when an exception is thrown by the main method?

Answer: When an exception is thrown by the main() method, Java Runtime terminates the program and print the exception message and stack trace in system console.

23. How do you ensure that N thread can access N resources without getting into the Deadlock situation?

Answer: In the case when N thread can access N resources without getting into a deadlock situation, here the key point in order. If we acquire resources in particular order and releases resources in reverse order then we can prevent the deadlock situation.

24. How to get the database server details in Java program?

Answer: For this, we can use DatabaseMetaData object to get the database server details. When the database connection is created successfully, we can get the metadata object by calling getMetaData() method. There are also many methods in DatabaseMetaData that we can use to know the product name, it's version and configuration details.

```
1. DatabaseMetaData metadata = con.getMetaData();
2. String dbProduct = metadata.getdatabaseProductName;
```

25. What is JDBC PreparedStatement?

Answer: JDBC PreparedStatement object represents a precompiled SQL statement. We can use its setter method to set the variables for the query. Since PreparedStatement is precompiled, it can then be used to efficiently execute this statement multiple times. PreparedStatement is a better choice than Statement because it automatically escapes the special characters and avoids SQL injection attacks.

26. What are the various access specifiers in Java Class?

Answer: There are four access specifiers in Java: public, protected, default (no access specifier), and private.

Public: The fields, methods, and classes that are declared public can be accessed by any other class.

Protected: The fields, methods, and classes that are declared protected can only be accessed by classes in the same package or subclasses in other packages.

Default (no access specifier): The fields, methods, and classes that have no access specifier are accessible only to classes in the same package.

Private: The fields, methods, and classes that are declared private are only accessible within the declared class itself.

27. What is a Object in Java?

Answer: An object in Java is a data structure that represents a real-world entity. In Java, an object can be a physical object like a car, or it can be an abstract concept like a mathematical formula.

Each object has its own data and behavior. Data is the information that the object contains, while behavior is the object's ability to perform certain actions.

Java objects are created using a class. A class is a template that defines the data and behavior of a particular type of object. Once a class has been defined, we can create objects of that class by using the new keyword.

28. State difference between Function overloading and Function overriding.

Polymorphism in Java is achieved by overriding and overloading.

Overloading	**Overriding**
A method overloading is a situation where a class contains more than one method named the same, but with different parameters.	A method override occurs when the superclass and the child class have the same names and arguments/parameters.
Different parameters are required for overloading.	Overriding requires the same parameters
Overloading takes place within a class.	A class can override another when the two classes share an IS-A (inheritance) relationship. To override, both the base class and its child class must exist.
Static binding is used for method overloading.	Dynamic binding is used for method overriding.
Compile-time polymorphism is demonstrated here.	Run-time polymorphism is demonstrated here.

Overloading	Overriding
If overloading breaks, a compiler error will appear, which can easily be fixed.	Overriding breaks can create serious problems since the effects can be seen at runtime.

29. Can you explain what is static in Java?

In Java, the static keyword is primarily used to manage memory. Static keywords can be applied to blocks, methods, variables, and nested classes. If you declare a variable or a method static, it belongs to the class, instead of an individual instance of the class. This keyword is used to refer to methods or variables that are the same across every instance of a class.

30. How is an abstract class different from an interface?

The interface and abstract classes are both special types of classes that contain only the declarations of methods rather than their implementations. Both serve to achieve abstraction by declaring abstract methods. Interfaces, however, are entirely different from abstract classes.

Interface Class	Abstract Class
Whenever an interface class is implemented, its subclass must define all of its abstract methods and provide their implementation.	Whenever an abstract class is inherited, its subclass isn't obligatory to define its abstract methods until the subclass uses the abstract methods.
Interfaces can only have abstract methods. As of Java 8, they can also have static and default methods.	An abstract class may contain both abstract methods and non-abstract methods.
Multiple inheritances is supported by the interface.	Multiple inheritances is not supported by abstract classes.
Interfaces can extend other Java interfaces only.	Abstract classes extend other Java classes and implement multiple Java interfaces.
Java interfaces have public members by default.	There can be different types of class members available in Java abstract classes, such as private and protected.
It is not possible to declare constructors or destructors in an interface.	Constructors and destructors can be declared in an abstract class.

Interface Class	Abstract Class
When declaring a method as abstract in an interface, the keyword "abstract" is optional.	To declare a method abstract in abstract classes, the keyword "abstract" is required.
Variables must only be final and static.	Static, non-static, and final, non-final variables can all be present in an abstract class.

31. Can Java applications be run without implementing the OOPs concept?

No, Java applications are unable to be run without implementing OOPs. Java applications rely on object-oriented programming concepts, and therefore cannot be implemented without them. Java is an object-oriented programming language, which means that every program must implement at least one class. Programs are usually composed of many classes and objects, which are instances of classes. C++, on the other hand, can be implemented without any OOPs, as it supports the structure programming model that is similar to the one found in C.

32.