# Top Java Interview Questions

1. What are the principle concepts of OOPS?

|  |  |
| --- | --- |
| **Principle Concepts of OOPs** | |
| **Concept** | **Description** |
| *Abstraction* | Abstraction is hiding the implementation details & only providing the functionality to the user |
| Encapsulation | Encapsulation is a mechanism of wrapping up the data and code together as a single unit |
| *Inheritance* | Inheritance is a process where one class acquires the properties of another |
| *Polymorphism* | Polymorphism is the ability of a variable, function or object to take multiple forms |

2. How does abstraction differ from encapsulation?

* Abstraction focuses on the interface of an object whereas Encapsulation prevents clients from seeing it’s inside view i.e. where the behavior of the abstraction is implemented.
* Abstraction solves the problem in the design side while Encapsulation is the Implementation.
* Encapsulation is the deliverables of Abstraction. Encapsulation barely talks about grouping up your abstraction to suit the developer needs.

3. What is an immutable object? How do you create one in Java?

Immutable objects are those whose state cannot be changed once they are created. Any modification will result in a new object e.g. String, Integer, and other wrapper class.

4. What are the differences between processes and threads?

* A process is an execution of a program whereas a Thread is a single execution sequence within a process. A process can contain multiple threads.
* Thread is at times called a lightweight process.

5. What is the purpose of garbage collection in Java? When is it used?

The purpose of garbage collection is to identify and discard the objects that are no longer needed by the application to facilitate the resources to be reclaimed and reused.

6. What is Polymorphism?

Polymorphism is briefly described as “one interface, many implementations”. Polymorphism is a characteristic of being able to assign a different meaning or usage to something in different contexts – specifically, to allow an entity such as a variable, a function, or an object to have more than one form. There are two types of polymorphism:

* Compile time polymorphism
* Run time polymorphism.

Compile time polymorphism is method overloading. Runtime time polymorphism is done using inheritance and interface.

7. In Java, what is the difference between method overloading and method overriding?

Method overloading in Java occurs when two or more methods in the same class have the exact same name, but different parameters. On the other hand, method overriding is defined as the case when a child class redefines the same method as a parent class. Overridden methods must have the same name, argument list, and return type. The overriding method may not limit the access of the method it overrides.

8. How do you differentiate abstract class from interface?

* Abstract keyword is used to create abstract class. Interface is the keyword for interfaces.
* Abstract classes can have method implementations whereas interfaces can’t.
* A class can extend only one abstract class but it can implement multiple interfaces.
* You can run abstract class if it has main () method but not an interface.

9. Can you override a private or static method in Java?

You cannot override a private or static method in Java. If you create a similar method with same return type and same method arguments in child class then it will hide the super class method; this is known as method hiding. Similarly, you cannot override a private method in sub class because it’s not accessible there. What you can do is create another private method with the same name in the child class.

10. What is Inheritance in Java?

Inheritance in Java is a mechanism in which one object acquires all the properties and behaviors of the parent object. The idea behind inheritance in Java is that you can create new classes building upon existing classes. When you inherit from an existing class, you can reuse methods and fields of parent class, and you can also add new methods and fields.

Inheritance represents the IS-A relationship, also known as parent-child relationship.

Inheritance is used for:

* Method Overriding (so runtime polymorphism can be achieved)
* Code Reusability

11. What is super in Java?

The super keyword in Java is a reference variable that is used to refer the immediate parent class object. Whenever you create the instance of subclass, an instance of parent class is created i.e. referred by super reference variable.

Java super Keyword is used to refer:

* Immediate parent class instance variable
* Immediate parent class constructor
* Immediate parent class method

12. What is constructor?

Constructor in Java is a special type of method that is used to initialize the object. It is invoked at the time of object creation. It constructs the values i.e. provides data for the object and that is why it is known as constructor. Rules for creating Java constructor:

* Constructor name must be same as its class name
* Constructor must have no explicit return type

Types of Java constructors:

* Default constructor (no-arg constructor)
* Parameterized constructor

13. What is the purpose of default constructor?

A constructor that has no parameter is known as default constructor.

Syntax of default constructor:

<class\_name>(){}

14. What kind of variables can a class consist?

A class consists of Local Variable, Instance Variables and Class Variables.

15. What is the default value of the local variables?

The local variables are not initialized to any default value; neither primitives nor object references.

16. What are the differences between path and classpath variables?

PATH is an environment variable used by the operating system to locate the executables. This is the reason we need to add the directory location in the PATH variable when we install Java or want any executable to be found by OS.

Classpath is specific to Java and used by Java executables to locate class files. We can provide the classpath location while running a Java application and it can be a directory, ZIP file or JAR file.

17. What does the ‘static’ keyword mean? Is it possible to override private or static method in Java?

The static keyword denotes that a member variable or method can be accessed, without requiring an instantiation of the class to which it belongs. You cannot override static methods in Java, because method overriding is based upon dynamic binding at runtime and static methods are statically binded at compile time. A static method is not associated with any instance of a class, so the concept is not applicable.

18. What are the differences between Heap and Stack Memory?

Major difference between Heap and Stack memory are:

* Heap memory is used by all the parts of the application whereas stack memory is used only by one thread of execution.
* When an object is created, it is always stored in the Heap space and stack memory contains the reference to it.
* Stack memory only contains local primitive variables and reference variables to objects in heap space.
* Memory management in stack is done in LIFO manner; it is more complex in Heap memory as it is used globally.

19. Explain different ways of creating a Thread. Which one would you prefer and why?

There are three ways of creating a Thread:

1) A class may extend the Thread class

2) A class may implement the Runnable interface

3) An application can use the Executor framework, in order to create a thread pool.

The Runnable interface is preferred, as it does not require an object to inherit the Thread class.

20. What is synchronization?

Synchronization refers to multi-threading. A synchronized block of code can be executed by only one thread at a time. As Java supports execution of multiple threads, two or more threads may access the same fields or objects. Synchronization is a process which keeps all concurrent threads in execution to be in sync. Synchronization avoids memory consistence errors caused due to inconsistent view of shared memory. When a method is declared as synchronized the thread holds the monitor for that method’s object. If another thread is executing the synchronized method the thread is blocked until that thread releases the monitor.

21. How can we achieve thread safety in Java?

The ways of achieving thread safety in Java are:

* Synchronization
* Atomic concurrent classes
* Implementing concurrent Lock interface
* Using volatile keyword
* Using immutable classes
* Thread safe classes.

22. What are the uses of synchronized keyword?

Synchronized keyword can be applied to static/non-static methods or a block of code. Only one thread at a time can access synchronized methods and if there are multiple threads trying to access the same method then other threads have to wait for the execution of method by one thread. Synchronized keyword provides a lock on the object and thus prevents race condition.

23. What are the differences between wait() and sleep()?

* Wait() is a method of Object class. Sleep() is a method of Thread class.
* Sleep() allows the thread to go to sleep state for x milliseconds. When a thread goes into sleep state it doesn’t release the lock.
* Wait() allows the thread to release the lock and go to suspended state. The thread is only active when a notify() or notifAll() method is called for the same object.

24. How does HashMap work in Java ?

A HashMap in Java stores key-value pairs. The HashMap requires a hash function and uses hashCode and equals methods in order to put and retrieve elements to and from the collection. When the put method is invoked, the HashMap calculates the hash value of the key and stores the pair in the appropriate index inside the collection. If the key exists then its value is updated with the new value. Some important characteristics of a HashMap are its capacity, its load factor and the threshold resizing.

25. What are the differences between String, StringBuffer and StringBuilder?

String is immutable and final in Java, so a new String is created whenever we do String manipulation. As String manipulations are resource consuming, Java provides two utility classes: StringBuffer and StringBuilder.

* StringBuffer and StringBuilder are mutable classes. StringBuffer operations are thread-safe and synchronized where StringBuilder operations are not thread-safe.
* StringBuffer is to be used when multiple threads are working on same String and StringBuilder in the single threaded environment.
* StringBuilder performance is faster when compared to StringBuffer because of no overhead of synchroniz