**📚 Complete Java Interview Question Bank**

**1️⃣ OOP Concepts**

**Basic Questions**

* What is OOP? Explain its 4 main principles with examples.
* Explain **inheritance** with an example. Can we achieve multiple inheritance in Java?
* What is **polymorphism**? Difference between compile-time & runtime polymorphism with examples.
* Explain **abstraction**. Difference between abstract class and interface.
* What is **encapsulation**? How is it implemented in Java?

**Conceptual / Deep-dive Questions**

* Where have you applied all four OOP principles in a real project?
* Can we achieve abstraction without using abstract classes?
* Explain how polymorphism helps in writing clean, scalable code.
* What happens internally when you override a method?

**Scenario-based Questions**

* Design a real-life example using all 4 OOP concepts.
* Given a base class Vehicle and subclasses Car and Bike, write code to demonstrate polymorphism.

**2️⃣ static, final, transient, volatile, and Immutability**

**Basic Questions**

* What is static in Java? Where can we use it (variables, methods, blocks, classes)?
* What is a static block? When is it executed?
* What does the final keyword do for variables, methods, and classes?
* What is the use of the transient keyword in serialization?
* What is volatile and where do we use it?
* Is String immutable? Why?

**Conceptual / Deep-dive Questions**

* Difference between static block, instance block, and constructor.
* Why should we make a class immutable?
* How do you create a custom immutable class?

**Scenario-based Questions**

* Write code showing how volatile solves visibility issues in multithreading.
* Write a custom immutable Employee class.

**3️⃣ Inner, Nested & Anonymous Classes**

* What is an inner class? Different types of inner classes in Java.
* How do you create an anonymous class?
* What are the advantages of using nested classes?
* Can an inner class access outer class private members?

**4️⃣ Exception Handling**

**Basic Questions**

* Difference between throw and throws.
* What is try-with-resources? How is it different from finally?
* When will the finally block not execute?
* Difference between **checked** and **unchecked** exceptions.

**Conceptual Questions**

* Explain exception hierarchy.
* Why should we create custom exceptions?

**Scenario-based Questions**

* Write code to create a custom exception class.
* Show multi-level exception catching using multiple catch blocks.

**5️⃣ Generics**

* What are generics? Why do we use them?
* Can we use other letters apart from T in generics?
* Write a generic class and a generic method.
* Difference between <? extends T> and <? super T>.

**6️⃣ Java Memory Model & Garbage Collection**

**Basic Questions**

* Difference between stack and heap memory.
* Explain young generation, old generation, and metaspace.
* What is a class loader?

**Conceptual Questions**

* Explain object life cycle in JVM.
* How does Garbage Collector work internally?
* Difference between CMS, G1, and Serial GC.

**Scenario-based Questions**

* Draw a diagram showing object allocation from creation to promotion.
* How do memory leaks happen in Java even with GC?

**7️⃣ Collections Framework**

**Basic Questions**

* Difference between ArrayList and LinkedList.
* Internal working of HashMap.
* Difference between HashSet, HashMap, and Hashtable.
* What is a TreeMap?

**Conceptual Questions**

* Explain load factor and threshold in HashMap.
* What is a fail-fast vs fail-safe iterator?
* What is CopyOnWriteArrayList and when to use it?
* Difference between synchronized collections and concurrent collections.

**Scenario-based Questions**

* Write code to group a list of employees by department using Collectors.groupingBy.
* Implement your own custom HashMap (basic version).

**8️⃣ Serialization & Deserialization**

* What is serialization? Why do we need it?
* Role of transient keyword in serialization.
* How can serialization be broken?
* What is serialization filtering (JDK 9)?
* Why do we sometimes use JSON/DB instead of serialization?

**9️⃣ Java 8 Features**

* What is a functional interface?
* Explain built-in functional interfaces like Predicate, Consumer, Supplier.
* What is a lambda expression?
* What is a method reference and constructor reference?
* Difference between map() and flatMap().
* Write code to use Collectors.joining, groupingBy, and summarizing.

**🔟 Design Patterns**

* Explain Singleton Design Pattern with code.
* What is double-checked locking in Singleton?
* How can we break Singleton and how do we prevent it (using enum)?
* Explain Builder, Factory, Abstract Factory, and Decorator design patterns with examples.

**1️⃣1️⃣ Strings**

* What is the String Constant Pool?
* Why are Strings immutable?
* Difference between String, StringBuffer, and StringBuilder.

**1️⃣2️⃣ Multithreading & Concurrency**

* Difference between process and thread.
* Explain thread life cycle.
* What is Executor Framework?
* Explain fixed thread pool and custom thread pool.
* Which thread pool is used in parallel streams?
* Difference between synchronized and Lock.
* What is CompletableFuture and its use cases?

**1️⃣3️⃣ File I/O & Date API**

* Explain NIO vs IO.
* How do you read/write files in Java 8+?
* What is the new Date/Time API and its advantages over java.util.Date?

**1️⃣4️⃣ GIT & Build Tools**

* Difference between merge and rebase.
* How do you resolve merge conflicts?
* Explain stash in Git.
* What is Maven/Gradle? Difference between them.

**1️⃣5️⃣ Testing**

* How to write JUnit test cases.
* What is Mockito?
* How to mock static, final, and private methods?
* How to assert exceptions?

**1️⃣6️⃣ Spring & Spring Boot**

* What is IoC and DI?
* Explain annotations: @Component, @Autowired, @Bean, @Configuration.
* What are bean scopes? Explain with real-world examples.
* Explain bean life cycle.
* What is Spring Boot auto-configuration?
* What is @SpringBootApplication?

**1️⃣7️⃣ Spring Security**

* Difference between Authentication and Authorization.
* Explain JWT and its structure.
* What are claims, audience, and expiration in JWT?
* What is OAuth2 flow?

**1️⃣8️⃣ JPA & Hibernate**

* Explain first-level and second-level cache.
* What is @Transactional? Explain propagation and isolation levels.
* What is an EntityManager?
* Explain relationships: OneToOne, OneToMany, ManyToMany.
* How do you write custom queries?

**1️⃣9️⃣ Microservices**

* What is API Gateway?
* Explain Service Registry and Discovery.
* Explain Circuit Breaker Pattern (Open, Half-Open, Closed).
* Difference between Orchestration and Choreography in Saga Pattern.
* What is Distributed Tracing? What are Span ID and Trace ID?
* Explain RestTemplate, WebClient, and Feign Client.

**2️⃣0️⃣ Agile & Dev Practices**

* Explain Agile ceremonies (Daily Standup, Sprint Planning, Retrospective).
* Difference between Scrum and Kanban.