## 1️⃣ Core Java – Basics & Fundamentals

1. What are the main features of Java that make it platform independent?
2. Explain JVM, JRE, and JDK. How do they interact?
3. What is the difference between == and equals()?
4. Explain primitive vs non-primitive data types.
5. What is immutability? Why is String immutable in Java?
6. Difference between final, finally, and finalize().
7. What is method overloading and method overriding?
8. What is the difference between ArrayList and LinkedList?
9. Explain access modifiers in Java with use cases.
10. What happens internally when you run a Java program?

---

## 2️⃣ Core Java – Algorithm Solving & DSA

1. Write a program to reverse a string without using built-in methods.
2. How do you check if a number is a palindrome?
3. Find the first non-repeating character in a string.
4. Difference between Stack and Queue with implementation examples.
5. Explain time and space complexity of linear search vs binary search.
6. How do you detect a loop in a linked list?
7. Write logic to find the second largest element in an array.
8. Explain recursion with a real-world example.
9. Implement a basic LRU cache using Java collections.
10. Explain HashMap internal working and how collisions are handled.

---

## 3️⃣ Core Java – Intermediate (Frequently Asked Interview Questions)

1. How does HashMap work internally? What happens when capacity exceeds threshold?
2. Difference between Comparable and Comparator with examples.
3. What are checked and unchecked exceptions? When do you use each?
4. Explain deep copy vs shallow copy.
5. How does garbage collection work in Java?
6. Difference between volatile and synchronized.

7. What is fail-fast and fail-safe iterator?
8. Explain Java memory model (Heap vs Stack).
9. What is serialization? How do you prevent serialization?
10. Explain the use of Optional class.

---

### 4️⃣ Core Java – Advanced

1. Explain class loading mechanism and different class loaders.
2. What are different garbage collectors (G1, CMS, ZGC)?
3. Explain Java Memory Leaks and how to identify them.
4. What is JIT compiler? How does it improve performance?
5. Explain concurrency utilities in java.util.concurrent.
6. Difference between ForkJoinPool and ExecutorService.
7. How does Java handle multi-threading at OS level?
8. Explain reactive programming in Java ecosystem.
9. What is Project Loom and virtual threads?
10. How does Java support microservices architecture?

---

### 5️⃣ Advanced Java

**A. Basics & Fundamentals (10 Questions)**

1. What is Advanced Java (J2EE)?
2. Difference between Servlet and JSP.
3. Explain request-response lifecycle in a servlet.
4. What is a web container?
5. Difference between GET and POST methods.
6. What is session management? Types?
7. What is JDBC and its architecture?
8. Explain connection pooling.
9. Difference between Statement and PreparedStatement.
10. What is MVC architecture?

---

**B. Intermediate**

1. How does Spring improve over traditional J2EE?
2. Explain filters and listeners in web applications.
3. What is ORM? How does Hibernate work?
4. Difference between eager and lazy loading.
5. What are transactions? ACID properties?
6. How does Spring handle dependency injection?
7. What is REST? Difference between REST and SOAP?
8. Explain HTTP status codes commonly used in APIs.
9. What is pagination and how do you implement it?
10. Explain exception handling in REST APIs.

---

**C. Advanced Java – Architecture & Ecosystem**

1. How does Spring Boot simplify microservices development?
2. Explain API Gateway and its role.
3. What is service discovery (Eureka/Consul)?
4. Explain distributed transactions and Saga pattern.
5. How do you handle security in microservices?
6. Explain OAuth2 and JWT.
7. How does Spring handle scalability?
8. What is circuit breaker pattern?
9. Explain event-driven architecture using Kafka/RabbitMQ.
10. How do you monitor Java applications in production?

---

**6️⃣ Design Patterns (Interview-Focused)**

1. What are design patterns and why are they important?
2. Explain Singleton pattern and its pitfalls.
3. Difference between Factory and Abstract Factory.
4. Explain Builder pattern with real-world example.
5. What is Observer pattern? Where is it used?
6. Difference between Strategy and State pattern.
7. Explain Proxy pattern with Spring example.

8. How is Decorator pattern different from inheritance?
9. Explain MVC and MVVM patterns.
10. Which design patterns are used internally in Spring?

---

### 7️⃣ Capstone Project – Real Interview Questions (Very Important)

These **decide selection/rejection** in senior interviews.

**A. Architecture & Design (10 Questions)**

1. Explain your project architecture end-to-end.
2. Why did you choose this architecture?
3. How did you handle scalability in your project?
4. How did you design database schema?
5. What were the biggest technical challenges?
6. How did you ensure loose coupling?
7. How did you handle versioning of APIs?
8. How did you manage configurations across environments?
9. How did you ensure high availability?
10. What would you redesign if given a chance?

---

**B. Coding & Implementation**

1. Explain a complex business logic you implemented.
2. How did you handle concurrency issues?
3. How did you optimize performance?
4. How did you handle exceptions globally?
5. How did you write unit and integration tests?
6. How did you ensure code quality?
7. How did you manage database transactions?
8. How did you handle large data volumes?
9. How did you secure APIs?
10. Show a piece of code you are most proud of.

---

**C. Production & DevOps**

1. How was your application deployed?
2. How did you monitor the application?
3. How did you handle production bugs?
4. What logging strategy did you use?
5. How did you manage secrets?
6. How did you handle rollback?
7. What CI/CD pipeline did you use?
8. How did you manage configuration changes?
9. How did you ensure zero downtime deployment?
10. How did you handle peak traffic?