**Detailed Topic List**

**1. JDK, JRE, JVM**

* **JVM (Java Virtual Machine)**
  + What is JVM and its role?
  + How JVM executes Java bytecode?
  + JVM architecture and components
  + JVM memory areas (Method Area, Heap, Stack, PC Registers, Native Method Stack)
  + Just-In-Time (JIT) compiler
  + Platform independence concept
* **JRE (Java Runtime Environment)**
  + Definition and components of JRE
  + Difference between JVM and JRE
  + What does JRE contain? (JVM + libraries + other files)
  + Role of JRE in running Java applications
* **JDK (Java Development Kit)**
  + What is JDK and what it includes?
  + Difference between JDK and JRE
  + Tools provided by JDK (javac, java, javadoc, jar, etc.)
  + JDK versions and features evolution

**2. Classloaders**

* What is a Classloader?
* Types of Classloaders
  + Bootstrap Classloader
  + Extension (Platform) Classloader
  + System (Application) Classloader
* Class loading process in JVM
* Parent delegation model of classloaders
* Custom classloaders (basic understanding)
* When and why ClassNotFoundException or NoClassDefFoundError occurs

**3. Memory Management: Stack and Heap**

* JVM memory model overview
* **Stack Memory**
  + What is stored in the stack? (Primitive data, references, method calls)
  + Stack frames and method call lifecycle
  + Stack Overflow error and causes
* **Heap Memory**
  + What is stored in the heap? (Objects, instance variables)
  + Heap structure (Young Generation, Old Generation, PermGen/Metaspace)
  + Garbage Collection overview
  + Difference between stack and heap
  + OutOfMemoryError scenarios

**4. Object-Oriented Programming (OOP) Concepts**

* Four main pillars of OOP:
  + **Encapsulation**
    - Data hiding, access modifiers (private, protected, public, default)
    - Getters and setters
  + **Inheritance**
    - IS-A relationship
    - extends keyword
    - Types of inheritance (single, multilevel, hierarchical)
    - super keyword
  + **Polymorphism**
    - Compile-time polymorphism (method overloading)
    - Runtime polymorphism (method overriding)
    - Upcasting and downcasting
  + **Abstraction**
    - Abstract classes and methods
    - Interfaces (Java 8+ default and static methods)
    - Difference between abstract class and interface
* Additional concepts:
  + Classes and Objects
  + Constructors and Constructor overloading
  + this keyword
  + final keyword (variables, methods, classes)
  + Static vs instance members
  + Inner classes and anonymous classes

**5. Method Overloading and Overriding**

* **Method Overloading**
  + Definition and rules
  + Compile-time polymorphism
  + Overloading with different parameter types, numbers, order
  + Constructor overloading
* **Method Overriding**
  + Definition and rules
  + Runtime polymorphism
  + Overriding vs overloading differences
  + Use of @Override annotation
  + Access modifier rules during overriding
  + Covariant return types
  + Calling overridden method with super

**6. Exception Handling**

* What are exceptions? Difference between errors and exceptions
* Checked vs unchecked exceptions
* Throwable class hierarchy: Throwable, Exception, Error, RuntimeException
* Try-catch-finally blocks
* Multiple catch blocks and multi-catch
* Throw vs throws keyword
* Custom exceptions (creating user-defined exceptions)
* Best practices in exception handling
* Common exceptions (NullPointerException, ClassCastException, IOException, etc.)
* Exception propagation and stack trace
* try-with-resources (Java 7+)

**7. Collections Framework**

* Overview of Collections API
* Collection interfaces hierarchy:
  + Collection, List, Set, Queue, Deque, Map
* **List Implementations**
  + ArrayList vs LinkedList (use cases, time complexities)
  + Vector and Stack (legacy classes)
  + CopyOnWriteArrayList (thread-safe)
* **Set Implementations**
  + HashSet vs LinkedHashSet vs TreeSet
  + Use cases and ordering behavior
  + How HashSet works (hashcode and equals contract)
* **Queue and Deque Implementations**
  + PriorityQueue
  + ArrayDeque
* **Map Implementations**
  + HashMap vs LinkedHashMap vs TreeMap
  + Hashtable (legacy)
  + ConcurrentHashMap basics
* Iterator and ListIterator interfaces
* How equals() and hashCode() affect Collections
* Differences between Comparable and Comparator interfaces
* Sorting collections with Collections.sort() and Comparator
* Synchronization and thread-safe collections