# Advanced Java Performance Tuning

Duration: 5 days / 40 hours
Prerequisites: Working Knowledge of Java Programming

**Module 1: JVM Memory Management Basics**

1. Basics of Java Memory Management
   - Why memory management is critical for performance
   - Memory leaks vs memory pressure
2. Java Memory Model: Heap and Stack
   - Stack memory behaviour
   - Heap allocation patterns
   - Impact on application latency
3. Components of JVM Memory
   - Young Generation
   - Old Generation
   - Metaspace
   - Code Cache (introductory)

**Module 2: Garbage Collection**

4. Introduction to Garbage Collection
   - What is GC and why it exists
   - Stop-the-World events
5. How Garbage Collection Works
   - Object allocation and promotion
   - Minor vs Major GC
6. Generational Garbage Collection
   - Eden, Survivor spaces
   - Object aging
   - Performance implications

**Module 3: Advanced GC & Multithreading Performance Garbage Collectors & Algorithms**

7. Types of Garbage Collectors
   - Serial GC – when to use
   - Parallel GC – throughput focus
   - CMS – low latency overview
   - G1 GC – region-based collection

8. Garbage Collection Algorithms
   - Mark-and-Sweep
   - Mark-and-Compact
   - Copying collectors

9. Understanding GC Metrics
   - Throughput
   - Latency

- Pause times
- Allocation rate

**Module 4: GC Tuning & Troubleshooting**

10. Tuning Garbage Collection
    - Heap sizing strategies
    - Young vs Old generation tuning
11. JVM Flags for GC Tuning
    - Commonly used GC flags
    - Heap and GC logging options
    - GC selection flags
12. Analysing Garbage Collection Logs
    - Reading GC logs
    - Identifying GC bottlenecks
    - Common GC tuning mistakes

**Module 5: Threads and Concurrency Fundamentals**

13. Basics of Java Threads
    - What is a thread?
    - OS threads vs Java threads
14. Creating Threads
    - Thread class vs Runnable
    - Performance implications of each approach
15. Thread Lifecycle and States
    - New, Runnable, Blocked
    - Waiting, Timed Waiting, Terminated
    - State transitions and performance impact
16. Starting and Joining Threads
    - Thread startup cost
    - Join and blocking issues

**Module 6: Multithreading & Concurrency Performance**

17. Understanding Multithreading
    - CPU cores and parallelism
    - Context switching overhead
18. Synchronization and Locks
    - Synchronized blocks
    - Lock contention
    - Deadlocks and performance degradation
19. Concurrency Best Practices
    - Reducing lock scope
    - Avoiding shared mutable state
    - Designing high-performance concurrent code

**Module 7: Java Virtual Threads**

20. Introduction to Virtual Threads

- What are Virtual Threads?
- Why Project Loom was introduced
- Problems with traditional thread-per-request model

21. Platform Threads vs Virtual Threads
    - OS thread limitations
    - Lightweight nature of virtual threads
    - Memory and scalability benefits
22. Creating and Using Virtual Threads
    - Creating virtual threads
    - Executors and virtual threads
    - Migrating existing code
23. Virtual Threads and Performance
    - Blocking I/O vs non-blocking I/O
    - Impact on throughput and latency
    - When virtual threads improve performance