Module 1: Introduction to Multithreading

1. **Welcome and Course Overview**
2. **Understanding Threads and Processes**
   - What is a thread?
   - Thread vs. Process
   - Advantages of Multithreading
3. **Creating and Managing Threads**
   - Extending the `Thread` class
   - Implementing the `Runnable` interface
   - Using lambda expressions (Java 8+)
   - Thread lifecycle: states and transitions

Module 2: Thread Control and Synchronization

4. **Basic Thread Operations**
   - `start()`, `run()`, `sleep()`, `yield()`, `join()`, `interrupt()`
   - Daemon threads
5. **Synchronization Mechanisms**
   - `synchronized` methods and blocks
   - Intrinsic locks and monitor concepts
6. **Advanced Synchronization Techniques**
   - `ReentrantLock` and `ReentrantReadWriteLock`
   - Using `Condition` objects
   - `volatile` keyword

Module 3: Concurrency Utilities and Advanced Topics

7. **Java Concurrency Utilities**
   - Overview of `java.util.concurrent`
   - Executors and thread pools (`ExecutorService`, `ScheduledExecutorService`)
   - `Future` and `Callable`
8. **Concurrent Collections**
   - `ConcurrentHashMap`
   - `CopyOnWriteArrayList`
   - `BlockingQueue` implementations (`ArrayBlockingQueue`, `LinkedBlockingQueue`)
9. **Atomic Variables and Locks**
   - `AtomicInteger`, `AtomicLong`, `AtomicReference`
   - `StampedLock`

Module 4: Advanced Thread Management

10. **Inter-Thread Communication**
    - `wait()`, `notify()`, `notifyAll()`
    - Producer-Consumer and Readers-Writers patterns
11. **Handling Thread Contention**
    - Deadlock, livelock, and starvation
    - Deadlock detection and avoidance
12. **Thread Local Storage**
    - `ThreadLocal` class

Module 5: High-Performance Concurrency

13. **Fork/Join Framework**
    - Introduction to Fork/Join
    - `RecursiveTask` and `RecursiveAction`
14. **Parallel Streams (Java 8+)**
    - Utilizing parallel streams
15. **Asynchronous Programming**
    - `CompletableFuture`
    - Combining and chaining futures

Module 6: Best Practices and Testing

16. **Best Practices for Multithreading**
    - Designing thread-safe classes
    - Immutable objects and thread confinement
17. **Testing Multithreaded Code**
    - Tools and frameworks
    - Writing reliable and repeatable tests
18. **Performance Tuning**
    - Profiling multithreaded applications
    - Identifying and fixing bottlenecks

Module 7: Practical Applications and Case Studies

19. **Real-World Scenarios**
    - Applying multithreading in real-world applications
    - Case studies of high-performance applications
20. **Concurrency Design Patterns**
    - Common design patterns
    - Practical implementations

Module 8: Advanced Topics and Emerging Trends

21. **Java Memory Model**
    - Happens-before relationship
    - Visibility guarantees
22. **Custom Synchronizers**
    - Building custom synchronizers with AbstractQueuedSynchronizer (AQS)
    - Implementing Semaphore, CountDownLatch, CyclicBarrier, and Phaser
23. **Concurrency in Modern Architectures**
    - Concurrency in microservices
    - Concurrency in cloud-based applications

Module 9: Capstone Project

25. **Capstone Project**
    - Designing and implementing a multithreaded application
    - Real-world project scenarios
    - Performance tuning and testing

Module 10: Virtual Thread (Java 21+ features)

26. **Project Loom**
    - What is a virtual thread?
    - Under the hood of virtual thread.
    - Subroutine vs Coroutine
    - Scoped Value
    - Structure concurrency.

Course Conclusion

Wrap-Up and Future Directions

- Recap of key concepts
- Further reading and advanced topics
- Final Q&A and course feedback