1. Operating systems concepts
   - Operating system services
   - Major issues studied in operating system development (processes, memory management, information protection and security, scheduling and resource management, and system structure)
   - Relevant issues in advanced operating systems (distributed OS, multiprocessor OS, database OS, real-time OS)

2. Processes, and concurrent process control
   - Process states and their transition
   - Representation of processes (PCB) and information maintained for each process
   - Lightweight (thread) vs. heavyweight processes
   - Interrupt processing and context switching
   - Concurrent processes and the need for their control
   - Process synchronization (mutual exclusion & general synchronization)
   - Mechanisms for providing mutual exclusion (hardware-based vs. software-based, busy waiting vs. non-busy waiting)
   - Software mechanisms with a single variable (semaphores, sequencers & event counts)
   - Mechanisms that allow multiple variables (OR, AND, NOT synchronization)
   - Strengths and weakness of each mechanism
   - Classical problems of synchronization and their solutions using the various mechanisms

3. Higher level concurrent programming mechanisms
   - Monitors
   - Serializers
   - Rendezvous implemented in Ada tasks
   - Open path expressions
   - Rationale for the development of high-level mechanisms
   - Strengths and weakness of each mechanism
   - Classical problems of synchronization and their solutions using the various mechanisms

4. Distributed concurrency control
   - Inherent problems in a distributed environment (lack of global clock & global memory)
   - Mechanisms to counter these problems (Lamport's logical clocks, vector clocks)
   - Applications of these mechanisms
     - Causal relation of events
     - Causal ordering of messages
     - Global state recording
     - Termination detection
   - Mutual exclusion algorithms in a distributed system
     - Non-token-based algorithms (Lamport's algorithm, The Ricart-Agrawala algorithm, Maekawa's algorithm)
     - Token-based algorithms (Suzuki-Kasami's broadcast algorithm, Singhal's heuristic algorithm, Raymond's tree-based algorithm)
   - Measure and comparison of performance (message traffic, synchronization delay, response time)

5. Deadlock
   - States & state transitions in terms of resource request/allocation
   - Necessary conditions for deadlock and their relevance to deadlock prevention
   - Sufficient condition for deadlock
   - Deadlock detection
     - Resource allocation graphs and graph reduction
     - Difficulty with deadlock detection in systems with reusable and consumable resources
     - Efficient deadlock detection algorithms for special cases
     - Resolution when deadlock is detected
   - Deadlock avoidance using the Banker's algorithm
6. Distributed deadlock detection
   - Difficulty with deadlock prevention and avoidance in distributed systems
   - Control organizations for distributed deadlock detection (centralized, distributed, hierarchical)
   - Deadlock detection
     - The possibility of detecting phantom deadlock
     - Algorithms with centralized control (completely centralized algorithm, the Ho-Ramamoorthy 2-phase & 1-phase algorithms)
     - Algorithms with distributed control (Obermarck's path-pushing algorithm, Chandy-Misra-Haas' edge-chasing algorithm)
     - Algorithms with hierarchical control (the Ho-Ramamoorthy algorithm)
     - Performance considerations (communication overhead, deadlock persistence time, storage overhead, processing overhead)
     - Deadlock resolution

7. Resource management & task scheduling
   - Modeling of scheduling problems
     - Nonpreemptive vs. preemptive scheduling
     - Dispatcher and context switching
     - Representation of schedules (Gantt charts)
     - Scheduling algorithms (first-come-first-served, shortest-job-first, priority, highest response ratio next, round-robin, multilevel-queue, multilevel-feedback-queue)
     - Performance measures (utilization, throughput, waiting time, response time, turnaround time)

8. Memory management
   - Management schemes, hardware/software support required and inherent problem of each
     - Single contiguous allocation (resident monitor approach)
     - Partitioned memory allocation (fixed partitions, variable partitions, fragmentation problems)
     - Paging
     - Segmentation
     - Combined systems (segmented paging, paged segmentation)
     - Virtual memory
   - Virtual memory implemented with paging
     - Hardware/software support
     - Instruction execution in a virtual memory system
     - Overhead in a virtual memory system
       - Page fault rate and the effective memory access time
       - Why use associative registers for page table
     - Page replacement
       - Algorithms (FIFO, OPT, LRU, LFU, MFU, etc.)
       - The FIFO anomaly
       - Implementation and hardware support required
       - Stack algorithms and calculation of cost as a function of available real memory size
       - The stack updating procedure
       - Local/global replacement
     - Means to speed up page swaps
     - Thrashing
       - Locality principle
       - Methods to reduce thrashing (working set model, page fault frequency strategy)
       - Page size considerations

REFERENCES
3. Shui Lam, CECS 526 Lecture Notes.