- Project related
  - Fork, thread, pipe, signal, socket
  - The basic concept of these functions
  - Under an example scenario, which ones are the best to use and "why"
    - May need to use multiple functions
    - There will be no question on thread specific behaviors, but combined with other functions
  - The behaviors of a program with these functions
    - Try out the behaviors if these functions are called differently
  - Projects 2 and 3
    - Basic designs
    - Thread synchronization and semaphore coding for Project 3
    - Different handling between the ready queue and terminal queue (how they are different and why)
    - Socket communication steps and select command

- Monitor solution
  - Using monitors to solve synchronization problems
  - Monitor implementation concepts and variations
    - Pros and cons for different implementations
  - Given a monitor code, understand whether it is correct and its behavior
  - Monitor states
    - State of the mutex for guarding the monitor
    - State of the condition variables and condition queues
    - State of other variables defined in the monitor
- Compare semaphore, monitor and lock
  - Analyze which ones would be most suitable for various situations

- Message passing primitives
  - Naming schemes, their powers and tradeoffs
  - BSBR versus NSBR versus NSNR, their powers and tradeoffs
  - Guarded communication
  - Socket and select communication

- Deadlock
  - Understand the difference between deadlock prevention, avoidance, and detection, understand their pros and cons
  - Follow the algorithms for deadlock prevention, avoidance, and detection
    - Linear ordering, avoid hold-and-wait, Banker's algorithm, wait-for graph
    - Understand why the algorithms would work
  - Given example new algorithms and decide
    - Which mechanisms they belong to
    - Whether it will work, and pros and cons
  - Given example scenarios and decide which algorithms will work better

- Memory management
  - Simple memory management algorithms,
    - First fit, best fit, … buddy
    - Simple paging
  - Physical address computation and access violations
  - Performance, fragmentation by the algorithms
- Virtual memory
  - Fully understand the concept of virtual memory
  - Understand demand paging, where pages are not brought in till it is needed and a page a program references may not be in memory
  - Memory hierarchy
    - Characteristics and average access latency
  - Page tables: 2-level, IPT, TLB
    - Understand the page table data structure and the pros and cons
    - Able to follow the scheme to perform addressing
    - What would be the access latency?
    - Which method is most suitable for a given scenario?
  - Swap space
    - What it is and its role in memory hierarchy
  - Memory access flow
    - Clearly know the entire flow
    - Understand when a page fault is raised and how it is handled
    - Check Project 2 Phase 1 and see how the registers will be impacted during a page fault
  - Working set management
    - What it is and the issues in setting the set size
    - Specific number of page faults with different working set size settings