Midterm 2 Study Guide

You are responsible for studying all content that can be found with the bellow slide decks. In general, you should be able to:

- Justify certain critical design decisions of operating systems (why use threads? Why use mutual exclusion?)
- Replicate exercises done in class (i.e., things I write in ink)

Below is an outline of topics covered so far.

Threads

- The difference between threads and processes
- Single threaded process model versus multi-threaded process model
- Why do programmers use threads
- Why are threads useful even on single-core systems
  - How interleaving typically works
- Thread states
- Types of thread implementation, and their relative costs/benefits
  - User-level
  - Kernel level
  - Combined
- Multi-core and multi-threading
  - Intuition behind Amdahl's law
  - Characteristics of programs that use multi-core programming effectively

Concurrency

- Interleaving versus overlapping
- Race conditions
- Role of OS in managing concurrency
- Kinds of concurrency between processes
- Potential problems because of concurrency
  - Define mutual exclusion and critical sections
  - Starvation
- Hardware support for mutual exclusion, and their pros and cons
  - Disabling interrupts
  - Compare and swap
- Software support for mutual exclusion, and their pros and cons
  - Semaphores
    - `n`
    - `semSignal`
    - `semWait`
    - Binary semaphores
    - Strong/weak semphores
    - How semaphores are used for mutual exclusion

        - How semaphores are based on hardware concurrency
    - Monitors
        - How monitors differ from semaphores
- Case studies:
    - The producer consumer case study for mutual exclusion
    - Message passing
        - Types
            - Blocking send, blocking receive
            - Non-blocking send, blocking receive
            - Non-blocking send, non-blocking receive
        - Direct vs indirect addressing

Deadlock and starvation

- Deadlock definition
- Joint process diagrams
- Safe/unsafe states
- Reusable versus consumable resources, and how deadlock can occur
- Resource allocation graphs, and identifying deadlocks in them
- The conditions leading to/causing deadlocks
    - Mutual exclusion
    - Hold-and-wait
    - No preemption
    - Circulation
- Dealing with deadlocks, and their pros and cons
    - Deadlock prevention
        - Disable mutual exclusion
        - Disable hold and wait
        - Allowing preemption
        - Disabling circular wait
    - Deadlock avoidance
        - Process and resource state matrices/vectors
        - Process initiation denial
        - Resource allocation denial
    - Deadlock detection
        - Deadlock detection algorithm
        - Deadlock recovery