OS and Architecture

- How to switch between user and kernel mode?
- How to switch among user processes?
- How to handle interrupts and system calls?
Processes and Threads

• What is a process?
• What is process control block? What is it used for? What information does it contain?
• What execution states can a process be in? What do they mean? What causes a process to change execution states?
• How does the OS keep track of processes?
• What is a context switch? What happens during a context switch? What causes a context switch to occur?
• What is the difference between a process and a thread?
• How are threads created? How is its state deleted?
Synchronization

- Why do we need to synchronize processes and threads?
- What is mutual exclusion?
- What is a critical section?
- What is a lock? How can we acquire and release a lock using interrupts/test&set instructions? What are the advantages and disadvantages of each approach?
- What is a semaphore? What are the different ways in which a semaphore may be used?
Synchronization

• Things you should be able to do:
  – Given code that uses locks, semaphores, monitors, you should be able to understand and explain whether or not it works. Does it guarantee mutual exclusion? Does it avoid starvation? Does it avoid deadlock?
  – You should be able to write pseudocode for a problem involving monitors or semaphores or locks like in the homeworks and sample midterms.
Deadlocks

- What is a deadlock? Difference from starvation
- Necessary conditions for a deadlock
- Deadlock detection, avoidance, prevention
- Resource allocation graph – deadlock detection
- Concept of a safe state
- Bankers algorithm
  - Problem solving with bankers algorithm.