Concurrency Bugs (Ch 32)
Discussion Questions

- What is an atomicity-violation bug?
  - non-atomic memory updates
  - ATM example
  - how do we fix them in general?
- What is an order-violation bug?
  - a specific ordering of operations is required, but not enforced
  - how is this generally fixed?
  - What are the four conditions that must hold for deadlock to occur? (explain in your own words)
- What are the differences in attitudes of deadlock prevention, avoidance, and detect/recover?
- How can we prevent circular waiting?
  - problems?
- How can we prevent hold-and-wait?
  - grab all locks at once, atomically
  - what are the problems with this approach?
- How can we prevent no preemption?
  - try lock & release all other locks if cannot obtain
  - how is this similar to hold-and-wait prevention technique?
- How can we prevent mutual exclusion?
  - wait-free data structures
  - is this a practical approach?
- What is livelock?
- Which of the four deadlock conditions does deadlock avoidance via scheduling attempt to remove?
  - Why is this impractical?
- Which approach do you think is the most common?

    - bury head in sand and hope they deadlock doesn’t happen
- What is the primary difference in the causes of non-deadlock bugs vs. deadlock bugs?
    - non-deadlocking => not enough control (locks, CV, etc…)
    - deadlock => using control wrong