CSCI 3753 / Final Study Guide

/ Notes

1. While I am making this study guide as detailed and in-depth as possible, and I am attempting to highlight the most relevant material to study, **I want to make clear that ANY material from lecture, programming assignments, recitation, reading assignments, and quizzes is fair game (unless specified otherwise below)**. That said, I will be focusing on the topics listed below.
2. Any pencil coding will NOT be nitpicked to ensure perfect C. In other words, no, I will not deduct a student for forgetting a semicolon, but you should stay as close to actual C code as possible.

/ Final Breakdown

The final is non-cumulative. That does not mean that material from the first half of the class will not appear, it just means that I will not be focusing the exam on it.

You will get 20 to 22 multiple choice questions and a scantron sheet. A significant portion of questions will be taken from and inspired by reading quizzes and lecture material. However, **please expect new questions as well. This will focus mostly on lecture and reading**, you may have questions regarding the programming assignments). 50% of the final grade.

You will also get 3-4 long answer questions. **Expect pencil coding**, and explaining concepts in detail. This will focus on programming assignments 4 and 5 as well as things like memory management, file system implementations, etc. **Please make sure to review: programming assignments 4 & 5 (including concepts), inodes, file systems, memory management algorithms and techniques, virtual memory, thrashing, thrashing minimization algorithms, networking protocols and layers, etc.** 50% of the midterm grade.

/ Topics to Skip

- Anything in the book about Solaris, Mach, or mobile operating systems
- Anything in the lectures and book about Windows… while this is important material, I will not be testing you over it on the midterm
- Enhanced clock algorithm
- Disk scheduling
- Flash file systems
- paravirtualization

/ Topics to Review

- The reading quiz questions
- Programming assignments 4-5 including all concepts needed to create a solution
- Fetch and execute cycle, how it relates to von Neumann architecture
- Memory hierarchy
- **Different caches - write-through, write-back, write-allocate, no-write-allocate and their uses**
- Cache replacement policies - LRU, etc.
- Memory Management
    - **MMU, memory mapping, segment and offset**
    - Physical vs. Logical (frames vs. pages)
    - Address binding at compile time
    - Address binding at load time
    - Address binding at run time
    - Swapping, including difficulties
    - Physical memory allocation and problems, fragmentation, etc.
    - **Paging!!!**
        - How MMU works with paging
        - Page size
        - Virtual memory, solves fragmentation
        - Page tables, how implemented, PTBR, page table types: basic, inverted, hierarchical (n-level), hashed page table,
        - TLB, methods for caching, how it relates to context switches, how lookups (hits and misses work)
- Memory and executable segmentation, names of segments
- Static vs. Dynamic linking, what they are and the difference between
- On-demand paging – benefits, advantages and problems, disadvantages… implementation
- **Page faults – including steps to load a page into RAM**
- **Page replacement policies**
    - modify and dirty bits
    - Page replacement algorithms: FIFO, OPT, LRU, LFU, MFU – the actual algorithms for, advantages vs. disadvantages
    - Belady's anomaly
    - Reference-bit LRU – Second-chance Clock Algorithm
    - How to improve page replacement performance?
    - Memory allocation policies
    - Local vs. Global allocation / replacement

- **Thrashing**
  - What it is, how it occurs
  - **Working set algorithm and technique** – what it is, how it works, understand it thoroughly
  - Page fault frequency technique for thrashing avoidance
  - Linux global page replacement (the Linux approach)
- **Memory Mapped Files** – what it is, how it works, uses, vs. standard file IO
- **File Systems**
  - How it works within the OS architecture
  - General idea of file systems and file metadata
  - System calls to alter/create/read/write files – how they work, steps
  - Tree structured directories
  - Links: symbolic vs. hard
  - File system mounting
  - 4 main file system components in memory
  - OFT
  - **File system implementations**
    - Strengths and weaknesses
    - Linked file allocation
    - **FAT**
    - Indexed allocation
    - Multilevel Indexed allocation
    - **UNIX Multilevel Indexed allocation - inode**
  - File allocation compared with process allocation
  - **Free space management**
    - Bit map
    - Linked list
    - Grouping
    - Counting / grouped linked list
  - File system fault recovery and fragility
    - **Log based recovery**
- Flash vs. spinning disk
- **RAID – what it means and EVERY RAID type, implementation types**
- **Networking**
  - How it works within an OS, kernel's network stack
  - **Protocol layers and their associated protocols in a traditional Internet stack**
  - Packetization
  - All the various layers and their purposes
  - **In particular, understand: IP, TCP, UDP, HTTP**
  - Network fault recovery – what causes packet loss?
  - Hub vs. switch
  - NFS

    - RPC
- Virtual machine
    - How a virtual machine works
    - What is a hypervisor
    - Shadow page tables
    - Intel VT-x

/ Hints on Long Answer Portion of Final

- Possible programming questions for the long-answer portion of the test:
    - PA4 and PA5 code questions – **EXPECT 1 FROM EACH PA**
        - LRU buffer (or something similar)
        - file system implementation from PA5
        - Working set algorithm
- While I will may ask you questions outside of these areas, the most important areas to review in detail are (this is where to start):
    - Reading Quiz questions
    - Programming Assignments
    - Memory management techniques
    - File System – in particular know UNIX inodes inside and out, be able to draw and explain everything about it
    - Memory management techniques vs. file system management techniques… compare them, what's next as memory becomes the file system?
    - Network protocol layers, as well as protocols that exist at every step
    - RAID levels
    - Memory mapped files and their uses