CSC 4320/6320: Operating Systems

Study Guide for Final Exam

Spring 2025

- 1 A4 size sheet, double-sided, handwritten by you is allowed in the exam.
- Please print your name and date clearly on top of your cheat sheet.
- MCQ: 40-50%
- Short Answer type/Coding/Code output: 50-55%

Some basic information:

- Final exam will be like Exam 1 and Exam 2. There will be two parts: MCQ and short answer type questions
- The syllabus (topics) after Exam 2 will constitute around 40% - 50% of the final exam points. These parts include:
    - Chapter 8 (Deadlock)
    - Chapter 9 (Main memory)
    - Chapter 10 (Virtual memory)
- The earlier chapters/topics (excluding chapter 7) will make up around 50-60% of the final exam.
- All the HW assignments, Exam 1, Exam 2 can be a very good source to understand example questions and the difficulty you may expect in the final.
- For MCQ part, you will need to go through the slides and take/understand key points of every topic.

More details:

chapters 1 – 10 (exclude chapter 7 from your exam syllabus)

questions will be similar to your exam 1 and exam 2

For example:

In exam 2, we had 22 MCQ questions, 10 short answer/computation type questions

For your final, you can expect:

MCQ: 20-24 questions

short answer/computation: 10-12 questions

Chapter 9: Main Memory - MCQ: 4 - 6, Short Answer/Computation: 2-4

Chapter 10: Virtual Memory - MCQ: 4 - 6, Short Answer/Computation: 2-4

For MCQ:1-2 questions per chapter from chapters 1-8 (exclude chapter 7)

For short answer/computation category: expect 1 question from every chapter

Short Answer/Computation: I strongly suggest reviewing all the homework assignments (HW1 – HW6, exclude HW4), Exam 1, and Exam 2.

Practice dynamic memory allocation algorithm: worst-fit, best-fit, first-fit

Practice banker's algorithm

Review resource allocation graph - deadlock conditions, etc.

Review CPU scheduling algorithms (computation and Gantt charts)

Expect some coding questions

Please also review other topics we studied throughout the semester. MCQ questions are conceptual, and hence you will need to go through the materials covered in this course.

Representative/Sample Questions

(Caution: Not a complete list of questions)

Chapter 10:

- Distinguish between logical memory and the physical memory.
- What is demand paging? How is it related to lazy swapper?
- What are the basic steps to handle a page fault?
- What is pure demand paging?
- To handle page fault during an instruction, we may need to restart an instruction. How an operating system deal with block of data transfer operation?
- Given the necessary information, calculate the EAT for demand paging
  For example:
  If memory access time is 250 nanoseconds and average page fault service time 10 milliseconds, the probability of page faults must be less _____ to keep the performance degradation less than 20%.
- What is Copy-on-Write mechanism? Distinguish between fork() and vfork().

- Given a reference string and the number of frames, calculate page faults using FIFO, Optimal, and LRU algorithms:
  For example: Suppose we have the following page accesses: 1 2 3 4 2 3 4 1 2 1 1 3 1 4 and that there are three frames within our system. Using the LRU replacement algorithm, what is the number of page faults for the given reference string?
- What is Belady's Anomaly?
- Distinguish fixed allocation and proportional allocation.
- What is the difference between global and local allocation of frames? What are their relative advantages and disadvantages?

Chapter 9:

- How are different processes protected from one another in main memory? Illustrate with a diagram.
- Define/differentiate compile time binding, load time, and execution time binding
- What is the difference between logical and physical address.
- Distinguish between static and dynamic linking
- Dynamic storage allocation: practice computational problem.
- What's the difference between external fragmentation and internal fragmentation.
- What is a memory compaction?
- What is paging and a page table?
- How is logical address translated to a physical address via a page table?
- Given process size, and the page size, compute internal fragmentation for the process.
- What is Translation Look-aside Buffer (TLB)? How does it work?
- Calculate Effective Access Time w.r.t TLB.
- What is the purpose of PTBR and PTLR?
- How does valid-invalid bit in a page table ensure memory protection?
- Practice computational problem for logical to physical address calculation.
- What is hierarchical paging? What is the issue of hierarchical paging with larger address space?
- What are hashed page table and inverted page table?
- Practice computational problem related to hierarchical paging and inverted page table.
- How to translate a logical address to a physical address in Intel 32 architecture?

Chapter 8:

- What are the four conditions required for deadlock occurrence?
- Distinguish deadlock and livelock,
- How to use resource allocation graph to identify deadlock.
- What are the methods for handling deadlock?
- What is the difference between deadlock prevention and deadlock avoidance?
- Define safe state, unsafe state, and deadlock state.
- Practice Banker's algorithm computational problem.
- Learn safety and resource-request algorithm (part of Banker's algorithm)

Chapter 7 (Excluded from the final exam syllabus)

Chapter 6:

- Critical section, requirements, Peterson's solution
- Race condition
- Requirements for solution of critical section problem
- Memory Barrier and Hardware instructions
- Mutex lock and semaphore solutions.

Chapter 5:

- Basic Concepts, Preemption/Non-preemption
- Practice all CPU scheduling algorithms: turnaround time, waiting time, etc.
- Estimation of burst-time, starvation, aging, load balancing.
- What are the relative advantages and disadvantages of different CPU scheduling algorithms.
- What are the scheduling criteria for CPU scheduling?
- Real-time CPU scheduling algorithms: Rate-monotonic and Earliest Deadline First algorithms.

Chapter 4:

Threads, benefits, concurrency and parallelism, data and task parallelism, Amdahl's law, user threads and kernel threads, their relationship, Implicit threading, thread pools, OpenMP, LWP

The following questions were suggested for Exam 1. So, they also should be part of your final exam syllabus.

Chapter 1:

1. What are the basic components of a computer system? Show their relative positions in a computer's logical organization.
2. Learn the functions of an operating system from different perspectives – users, workstations, mobile devices, embedded systems.
3. What is a system program and an application program?
4. Define Operating System.
5. You must learn concepts from the slides. Also read questions from slides.
6. How is a computer system generally organized? Show/describe.
7. What is an interrupt vector? Describe what happens when an interrupt happens.
8. The timing diagram of an interrupt.
9. Maskable and non-maskable interrupt.
10. Difference between synchronous and asynchronous I/O.
11. Learn the storage device hierarchy.
12. How does DMA work? When is DMA preferred over Interrupt? State some applications.
13. What is multiprocessing? Distinguish between symmetric and asymmetric multiprocessing.
14. Learn about CPU, Processor, Core, Multicore, and Multiprocessor
15. Multicore systems vs. Multiprocessor systems---Which one is better? Why?
16. What is NUMA System?
17. Does a dual-core system have its own cache memory? Why or why not?
18. How are clustered systems different from multicore systems?
19. Distinguish between multiprogramming and multitasking.
20. What are the benefits of multiprogramming over sequential processing?
21. How time-sharing systems run multiple processes currently even with a single processor?
22. Distinguish between user model and privileged (kernel) mode. Mention some operations that can be done in user-mode and in kernel-mode.
23. Process management and memory management activities
24. What is Caching? How does cache memory work?
25. Emulation vs. Virtualization. What are their use cases?
26. Name some kernel data structure and learn about them.

Chapter 2:

1. OS services – Mention some OS services for example.
2. What is a system call? Mention at least five system calls and tell their functions.
3. How would you access/use system calls in your program?

4. What are the different ways to pass parameters to system calls? Which method would you choose?
5. Learn the functions of major system calls like fork (), exit (), wait (), open (), read (), write (), close (), getpid(), sleep(), ioctl(), etc.
6. How do fork () and exec () system calls work together to run a new process?
7. Mention some system programs that provide some system services.
8. Linkers and loaders – what do they do?
9. Object code vs. Executable code
10. Why are applications OS specific?
11. Learn about different OS structure – simple/monolithic, layered, and microkernel. Benefits and challenges of each structure.
12. What are modules (loadable kernel modules)?
13. Have some ideas about the structures of macOS and iOS, Darwin, Android.
14. How to use fork () to create processes? Practice small programs.

Chapter 3

1. Program and process distinction.
2. Process layout in memory.
3. Process control blocks and process states. Define each state. Show state transition diagram.
4. What is context switching? Why is it a pure overhead?
5. Practice fork system calls.
6. Practice from HW2
7. Combine fork and execl to run an executable hello
8. Practice zombie and orphan process and (code).
9. How does a parent process wait for the child for its completion and the status?
10. Inter-process communication model – shared memory model and message passing model.
11. Race condition. Why does it happen? What are the ways you can think of to resolve the problem?
12. Problems with producer-consumer model.
13. Why there is no race condition in the circular array solution where one location is unused?
14. What is blocking/unblocking message passing?