CS365 Study Guide (I reserve the right to interpret if a question in is covered by this guide. Still, only about 90% of questions are covered by this guide.)

Chapter 1

- Know what is an operating system.
- Be familiar with the figure on the slide 1.6
- Know the steps to start up a computer.
- Understand the concept of interrupt: What is an interrupt, What does it do, Interrupt handling.
- Know storage hierarchy: Caching, Device driver, DMA,…
- Know computer system architecture: Asymmetric Multiprocessing, Symmetric Multiprocessing, Multicore, Cluster

Chapter 2

- Be able to describe the services an operating system provides: User interface, Program execution, I/O operations, File-system manipulation, Communications, Error detection, Resource allocation, Accounting, Protection and security. Figure on the slide 2.7
- Know system calls: What are system calls, How is an system call implemented (Figure on the slide 2.17), Parameter passing, Three common API to system call.
- Know the difference between "policy" and "mechanism".
- Know the various operating system structures: Monolithic, Layered, Microkernel, Loadable.
- Know some common operating systems' structures: Linux, Windows, Mac OS X.

Chapter 3

- Know process concepts: What is a process, Process components in memory,
- Understand the transmission of process states. Figure on the slide 3.8.
- Know PCB and at least 5 components of PCB.
- Understand the queues in process scheduling, Figure on the slide 3.15.
- Know the differences of various process schedulers: short-term, long-term, medium-term.
- Understand context switch.
- Know the basic APIs: fork(), exec(), exit(), abort(), wait(), getPid(), and their corresponding windows APIs.
- Understand interprocess communication: shared memory, message passing
- Communications in Client-Server Systems.

Chapter 4

- Understand thread concepts: What is a thread, Why is it light weight.
- Know the differences between parallelism and concurrency.
- Know the three multithreading models: Many-to-One, One-to-One, Many-to-Many (two level). Know at least one example system in each model.
- Know the three primary thread libraries and the basic operations: pthread_create(), pthread_join(),pthread_exit(), and their corresponding operations in windows library and java library if there are.
- Understand threading issues: semantics of fork(), signal handling, thread cancellation, thread local storage, scheduler activations.
- Windows threads and Linux threads

Chapter 5

- Know basic concepts: Process execution cycle, short-term scheduler, preemptive scheduling, non-preemtive scheduling, dispatch latency,
- Know scheduling criteria: CPU utilization, Throughput, Turnaround time, Waiting time, Response time
- Know how to calculate a schedule under above criteria
- Understand scheduling algorithms: FCFS, SJF, Priority, RR, Multilevel (feedback) Queue
- Know how to draw a Gantt Chart of a schedule under above scheduling algorithms
- Understand starvation and its solution
- Know concepts: asymmetric multiprocessing, symmetric multiprocessing (SMP)
- Understand processor affinity, load balancing, push migration, pull migration

Chapter 6

- Know basic concepts: race condition, atomic operation, busy waiting, spinlock,

- Understand process synchronization, why it is necessary
- Understand what is critical section problem.
- Know the three requirements for the solution of critical section problem, can prove whether a solution satisfies the three CS requirements
- Understand Peterson's solution and hardware solution, and know why they are not used
- Understand Mutexs, Semaphore, and Monitor, know the differences between them
- Know deadlock, starvation, and priority inversion problem in semaphore usage
- Understand the three classical synchronization problems: Bounded-Buffer Problem, Readers and Writers Problem, and Dining-Philosopher Problem
- Can fill blanks if a partial solution of the above problems are given.

Chapter 7

- Know basic concepts: system resources, request edge, assignment edge, claim edge, safe state, unsafe state,
- Understand the four conditions for deadlock
- Know resource-allocation graph and how to draw a resource-allocation graph
- Know how to check if there is a deadlock in a resource-allocation graph (single instance resource, multiple instances resource)
- Know methods to handle deadlocks: prevention, avoidance, detection and recovery, or ignore
- Understand deadlock prevention: violate one deadlock condition, ordering lock
- Understand deadlock avoidance, slide 7.21
- Know avoidance algorithms: resource-allocation graph and banker's algorithm, Can use these algorithms given a system with processes
- Understand deadlock detection: wait-for graph
- Know detection algorithm and know how to use it
- Understand deadlock recovery: select a victim and rollback

Chapter 8

- Know basic concepts: logical memory, physical memory, MMU, page, frame, and so on
- Understand contiguous memory allocation, noncontiguous memory allocation, external fragmentation, internal fragmentation
- Know how to convert from the logical address to physical address for segmentation, paging, hierarchical paging
- Know how to convert a address space to page representation

Chapter 17

- Know basic concepts: distributed system, network system, LAN, WAN, and so on
- Know how Ethernet works, how token ring works
- Understand problems and solutions in communication: addressing problem, routing problem, connection problem
- Know OSI 7-layers and TCP/IP protocol stack