Rules for Midterm, 150 points: Work alone and independently; open books, open internet, open notes. Place your name onto your midterm. Each question is worth 10 points max. Answer via single words or short phrases where feasible, minimize full sentences.
Mario Palacios
CSC 139 – Section 06

1. Describe function, method, and restrictions of Direct Memory Access (***DMA***)
- Function → used for high-speed IO devices able to transmit information at close to memory speeds
- Method → transfers blocks of data from buffer storage directly to or from main memory without CPU intervention
- Restrictions → data cannot be referenced by instructions before DMA is complete

2. Give an example of ***hard real-time problem***. What makes it *real-time,* what *hard*?
- An example is air traffic control
  - Real Time → must support priority-based scheduling
  - Hard → must be able to meet deadlines for planes landing

3. In languages like C++ the formal parameter "argc" of ***main()*** specifies the number of elements in "argv". No such specification is provided for "envp"; why?
- There is no specification for envp because it hold all the variables as a string constant, and is never zero

4. List the 4 conditions that can cause a ***deadlock*** for processes P1 and P2.
- Mutual exclusion
- Hold and wait
- No preemption
- Circular wait

5. Under which conditions does ***demand-paged VMM*** work well? Characterize a piece of "evil" software written to constantly cause page faults. How poorly will such "evil" SW run?
- performs well when there is good locality and number of page misses is less than address access
- evil software → fills working set with zero filled pages or will access a demand zero page

6. Explain what is essential to ***genuine interrupts***: Who initiates them? Why needed? How can a user infer from code, where and when interrupts arise?
- OS initiates the interrupts
- We need them because it gives user better control over the computer, and deals with processes immediately or via priority
- With the use of object codes a user can see where and when interrupts arise

7. The so-called ***Belady's Anomaly*** was discussed. What is the context? What is abnormal? What was plausible to expect instead of this anomaly?
- Context → LRU and optimal Page Replacement Algorithms
- Abnormal → in FIFO algorithm page faults increased with increase in number of frames
- Expected Outcome → referencing to the number of page faults being reduced if number of frames increase

8. Explain high-level the key points of ***thread***. Outline what is a "hyper-thread".
- Runs concurrently, each thread has its own stack, register set, condition codes, and control over full static address space

- Hyper-thread → is simultaneous multi-threading

> allows for multiple threads to run concurrerntly

> faster than a single μP that is shared concurrently with multiple threads

> slower than executing on a genuine Multi Core μP executing multiple threads or even multiple processes

> has at least one real, complete μP, with all ALU functions, registers, condition codes in silicon & access to memory

9. Define ***page fault***, and what causes them. Which OS actions are required in case of page faults? What run-time cost (relative to memory access cost) is associated with a page fault? What is "thrashing"?

- Page Fault → Logical address references a page that is not resident
- OS must have information to decide, find free frame, swap page into frame via scheduled dis operation, reset tables
- In reference to memory set present bit = p
- Thrashing → process is busy swapping pages in and out

10. Define ***Working Set***. What is the context? What are consequences of incorrect working sets?

- Working Set → set of available page frames necessary to enable fast program execution in a paged VMM system
- Context → Sufficiently large working set else thrashing will occur
- If incorrect it will either encompass entire locality, encompass several localities or will encompass entire program

11. Describe ***Preemptive Scheduler*** in an OS. Briefly state assumptions, goal, method, effect.

- Assumptions → Switches from running state (waiting state) to ready state
- Goal → excerise fair scheduling, large processes allow for small processes to wait to be executed
- Method → time slicing, grant CPU cycles in defined burst

12. What is ***Dispatch Latency*** in Operating Systems?

- time is takes the dispatcher to stop one process and start another process

13. Explain the ***system()*** function in C++ like programming environments. Which library is needed? What is the meaning of: ***system("man system")***?

- passes sole string parameter to Linux shell, as if string had been typed via shell command or via multiple shell commands
- available via library #include <stdlib.h>
- system("man system") → prints out the UNIX man page

14. Given a Resource Allocation Graph G with processes Pi and Resources Rj. G shows a particular situation. Under which condition of G is it clear that a deadlock cannot occur?

- Under the condition that the graph contains no cycles, that a deadlock will not occur

15. What are the key functions of ***kill(), fork(), exec()*** in a Linux runtime environment?

- Kill() → abort a process in UNIX whose PID is known type command *kill -9*
- Fork() → creates a new process by cloning the current process, to compile must use #include <unistd.h>
- Exec() → replace running process with new, including all threads