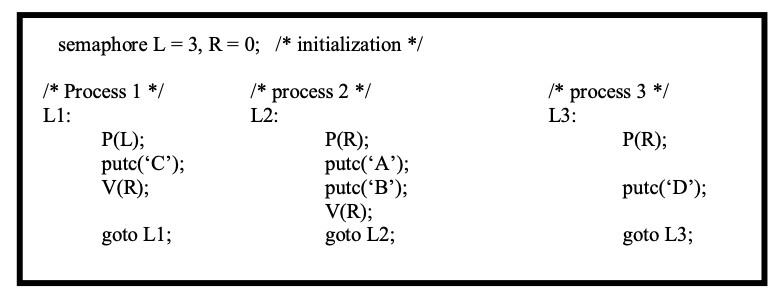
**Ch.6 – Process Synchronization**

* Race condition
* Scheduling algorithms
  + FCFS, SJF, SRTF, RR, Priority
* Time quantum
  + CPU bound vs. I/O bound
* Multilevel Queues and Multilevel Feedback Queues
* Linux O(1) and CFS scheduler
* In the following code, three processes produce output using the routine “putc” and synchronize using two semaphores “L” and “R.” 

a) How many D’s are printed when this set of processes runs? **3**

b) What is the smallest number of A’s that might be printed when this set of processes runs? **0**

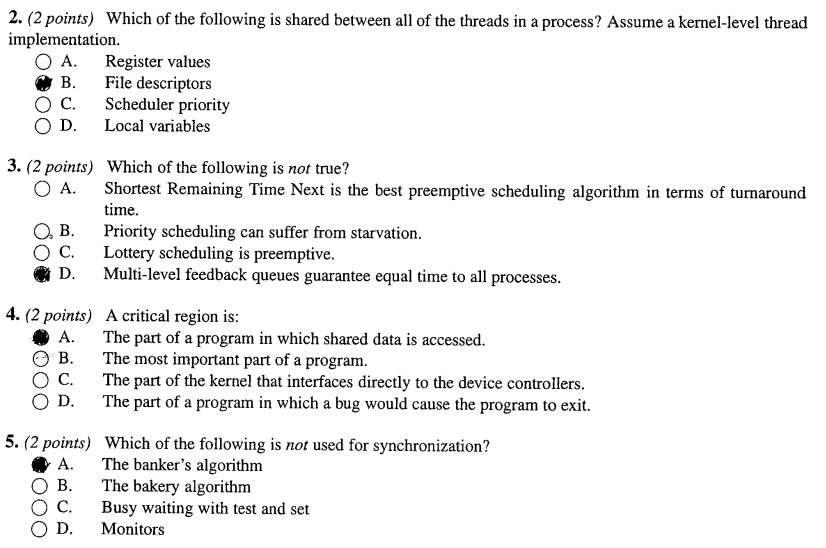
c) Is CABABDDCABCABD a possible output sequence when this set of processes runs? **no**

d) Is CABACDBCABDD a possible output sequence when this set of processes runs? **yes**

Ch. 8 – Deadlocks

* Definition, four necessary conditions
* Resource-allocation graph
* Three approaches to handle deadlocks
* Deadlock prevention
  + How to deny one of the deadlock conditions
  + Pros and cons
* Deadlock avoidance
  + Resource allocation states, safe and unsafe states, safe sequence,
  + Banker’s algorithm, safety test, pros and cons
* **T** The circular-wait condition for deadlock implies the hold-and-wait condition.
* **T** Deadlock can never occur if no is allowed to hold a resource requesting another resource.
* **T** Even if a system is in an unsafe state, it is possible for the to complete their execution without entering a deadlock state
* **F** In a virtual memory system, a virtual address and a physical address must be the same size
* Deadlock detection and recovery
  + From resource allocation graph to wait-for graph
  + Detection and recovery issues
  + Pros and cons.

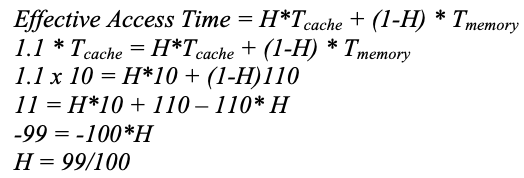
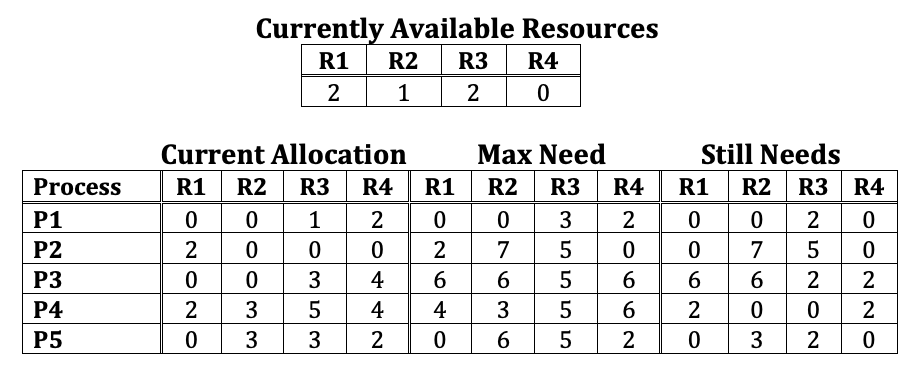
Ch. 8 – Memory Management

* Logical address space
  + Base and limit registers
* Compiled code addresses bind to relocatable addresses
  + Compile time, load time, execution time
* Page-based vs. segment-based memory management
* Hardware support
  + MMU
  + registers
  + cache
* Contiguous memory allocation
  + Storage allocation problem and its solutions
  + Internal fragmentation
  + External fragmentation
* A race condition is when ( C ) **the correctness of the code depends upon the timing of the execution**
* The Producer-Consumer Problem is related to allocation of resources
* 
* A system has 3 processes sharing 4 resources. If each process needs a maximum of 2 units, then deadlock **can never occur.**
* Which of the following is not true about segment based memory management? **Segment length must be a multiple of the page size.**
* Physical memory
  + frames
* Logical memory
  + pages
* Page tables
  + Page number, page offset
  + Page table entry, page table size
* Translation Look-aside Buffer (TLB)
* Valid and invalid bits
* Multilevel page tables
* Inverted page tables
* Hashed page tables

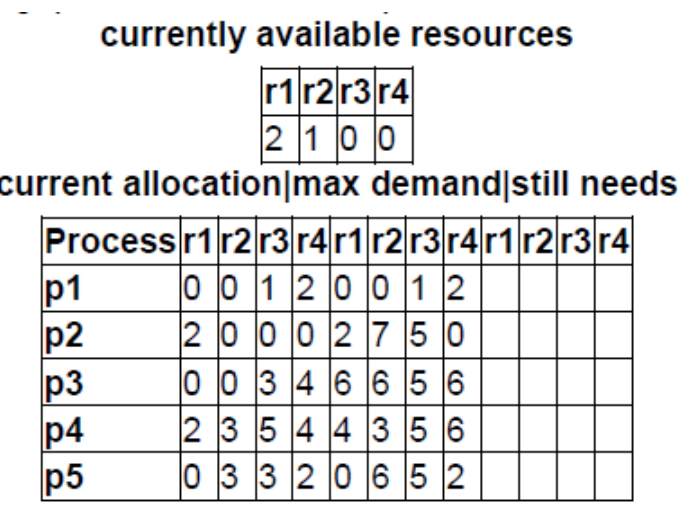
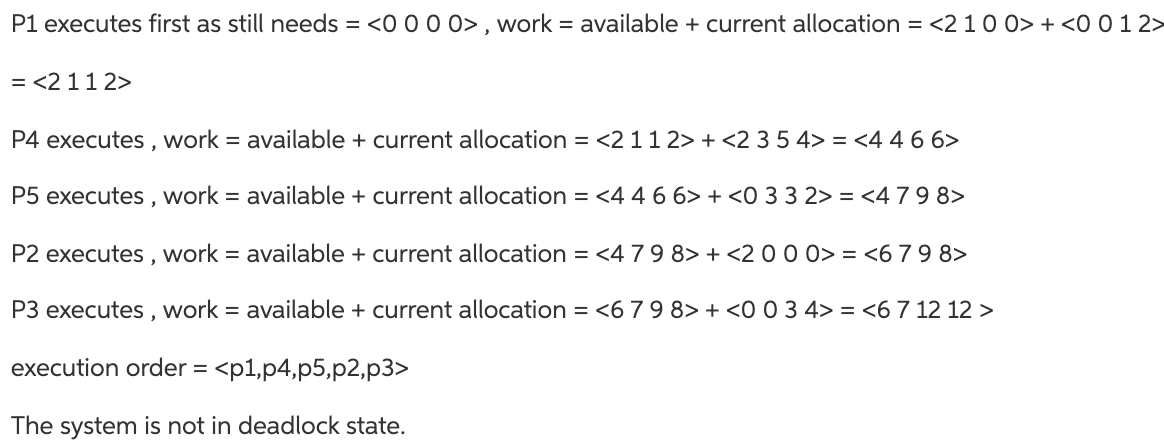
Ch. 9 – Memory Management

* Logical address space
  + Base and limit registers
* Compiled code addresses bind to relocatable addresses
  + Compile time, load time, execution time
* Page-based vs. segment-based memory management
* Hardware support
  + MMU
  + registers
  + cache
* Contiguous memory allocation
  + Storage allocation problem and its solutions \*
    - first-fit, best-fit, and worst-fit
  + Internal fragmentation
  + External fragmentation
* Physical memory
  + frames
* Logical memory
  + pages
* Page tables \*
  + Page number, page offset
  + Page table entry, page table size
* Translation Look-aside Buffer (TLB)
* Valid and invalid bits
* Multilevel page tables
* Inverted page tables
* Hashed page tables
* Bounded waiting implies that there exists a bound on the number Of times a process is allowed to enter its critical section **after a process has made a request to enter its critical section and before the request is granted.**

Ch. 10 – Virtual Memory

* Virtual memory
  + Benefits
* Page fault
  + Generation and handling
  + Performance: effective access time
* Kernel memory allocation
  + Buddy allocator vs. slab allocator
* Demand paging
  + Page replacement policies: FIFO, MIN (a.k.a. OPT), LRU, etc
* A multi-level page table is preferred in comparison to n. single-level page table for translating virtual address to physical address **it helps to reduce the size of page table needed to implement the virtual address space of a process.**
* Which Of the following is not true Of virtual memory **It requires hardware support**
* External fragmentation will not occur when  **No matter which algorithm is used. it will always occur**
* The purpose of A TLB is **to cache page translation information.**
* Which page replacement algorithm suffers from Belady's anomaly? **First In First out (FIFO)**
* What is the difference between the unit and signal operations of a semaphore and those of a condition variable (of a monitor)? **A call to the condition variable operation Wait() will always cause the calling thread to block (wait). The semaphore P() is a test-and-maybe-wait operation: only if the semaphore is not positive will the thread be forced to wait.**
* What is the difference between deadlock prevention and deadlock avoidance? What category does Banker's algorithm fall in and why? **Prevention: Preventing deadlocks by constraining how requests for resources can be made in the system and how they are handled. Avoidance: The system dynamically considers every request and decides whether it is safe to grant it at this point. Banker’s algorithm falls under avoidance because it makes an “s-state” check to test for possible activities before allowing to continue.**
* Consider a memory system with a cache access time of 10ns and a memory time access time of 110ns – assume the memory includes the time to cheek the cache. If the effective time access time is greater than the cache access time. what is the hit ratio H? 
* **T** Hashed page tables are particularly useful for processes with sparse address spaces
* **F** Segmentation avoids external fragmentation.
* **T** The buddy system for allocating kernel memory is very likely to cause fragmentation within the allocated segments
* **F** A 32-bit logical address with 8 KB page size will have I ,OOO entries in a conventional page. table
* **T** A page fault must be preceded by a TLB miss
* **T** Stack algorithms can never exhibit Belady's anomaly.
* Consider the following snapshot of a system with five processes (P1, P2, P3, P4, P5) and four resources (R1, R2, R3, R4). There are no current outstanding queued unsatisfied requests. 

1. Fill in the contents of the matrix Need for each process **Still needs=Max-Current**
2. Is this system currently deadlocked, or can any process become deadlocked? Why or why not? If not deadlocked. give an execution order. **Not deadlocked and will not become deadlocked. Using the Banker’s algorithm: P1, P4, P5, P2, P3.**

**Not same problem->** ** Compare need<= available and if True, add current alloc + available**

1. If a request from a process P1 arrives for (0, 4, 2, 0). can the request be immediately granted? Why or why not? If yes, show execution order. **No, the request is invalid as it would exceed the maximum need that P1 specified at the time of its creation.**
2. If a request from a process P2 arrives for (0, 1, 2, 0), should the request be immediately granted? Why or why not? If yes, show an execution order. **No, the request is valid but if granted, the resulting Currently Available Resources would be (2, 0, 0, 0) and there is no sequence of process executions that would allow the completion of all processes. This is an UNSAFE state.**

