OPERATING SYSTEMS (COMP 310 / ECSE 427) – Midterm – November 14, 2018: 6:05pm
Duration: 60 mins

Name: Answers - **Student ID:**

Part I – Answer **all 10 questions – 5 points each**

1. In a particular CPU architecture, the VMM is implemented using binary translation. Can we revise that implementation to use trap-and-emulate? Briefly explain your answer.

No. Binary translation is used because sensitive instructions $\subseteq$ privileged instructions.

2. List one advantage and one disadvantage of para virtualization.

ADV: Para virtualization can result in light weight virtualization. DISADV: Guest kernel is modified.

3. Give an example scenario where busy waiting would be a better option than sleep/wakeup.

Tiny critical sections with few contending threads.
A critical section for modifying a shared variable by few threads that are not likely to try to enter the critical section at the same time.

4. Implement a stack data structure using monitor. Show the pseudo code for the pop() and push() operations.

infinite storage – never overflowing it -

```
Monitor Stack {
    pool_t pool;
    condvar empty;
    push (item i) {
        pool.fadd(i);
        empty.signal();
    }
    pop () {
        if (pool.size() == 0)
            empty.wait();
        return pool.frm();
    }
}
```

pool_t has two operations fadd() put it item at the front. frm() remove front item.

5. Briefly explain what is meant by priority inversion. Suppose a system with two processes are deadlocked due to priority inversion. What would be a simple strategy to resolve the deadlock problem without killing any process?

A high priority process H is unable to take a lock because a low priority process L is holding it and has been evicted from the CPU by a priority based scheduler. SOLUTION: Temporarily boost the priority of the low priority so it will run and release the lock.

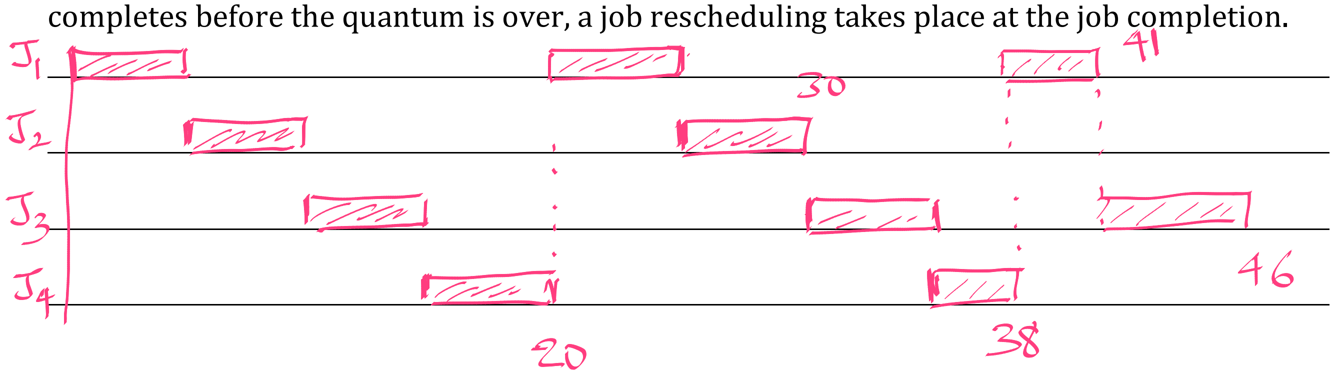
6. We studied "taking turns" (with one shared "turn" variable) as a possible approach to implementing critical sections to evade race conditions. Why was that approach rejected? What requirement (in terms of race condition evasion) was not met by this approach? Very briefly explain your answer.

A process outside its critical section can prevent another process from getting into the critical section. This violates a requirement for the critical section solution.

7. Describe three deadlock prevention schemes.

- Resource ordering so that all processes request in the same order
- Requesting all resources at the same time
- Preempts processes holding onto resources

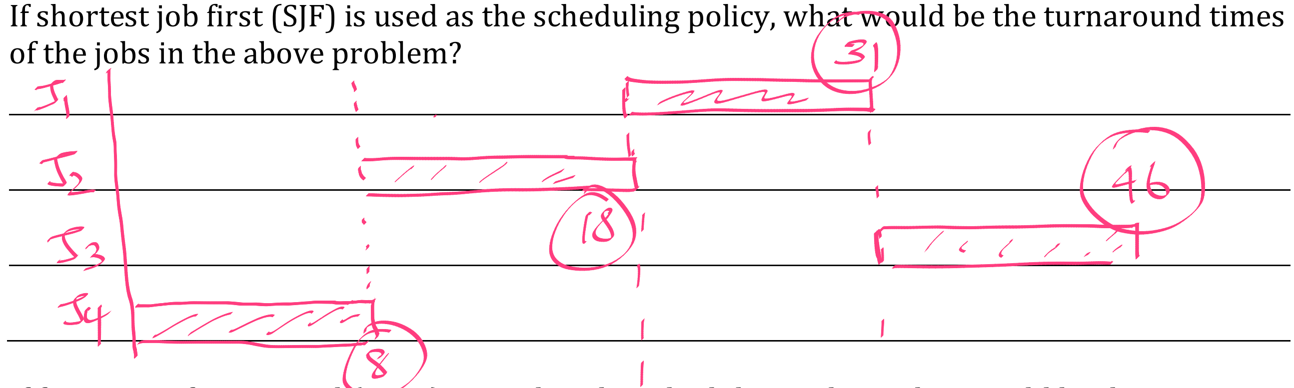
8. Consider four jobs that arrive at the scheduler at time 0 with the following CPU burst time requirements: $J_1(13)$, $J_2(10)$, $J_3(15)$, $J_4(8)$. If round-robin (RR with quantum size 5) is used as the CPU scheduling policy, what would be the turnaround times of the different jobs? If a job completes before the quantum is over, a job rescheduling takes place at the job completion.



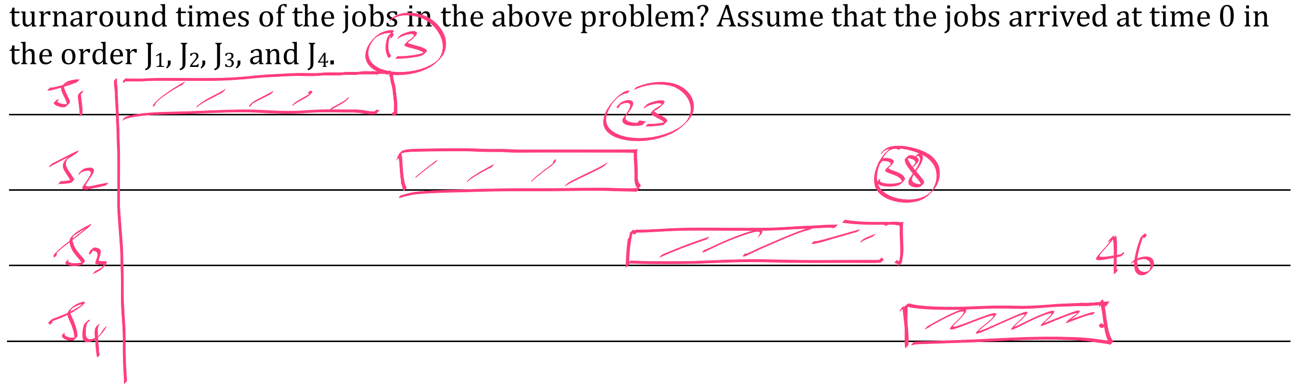
J1 – 41   J3 – 46

J2 – 30   J4 – 38

9. If shortest job first (SJF) is used as the scheduling policy, what would be the turnaround times of the jobs in the above problem?



10. If first-come first-served (FCFS) is used as the scheduling policy, what would be the turnaround times of the jobs in the above problem? Assume that the jobs arrived at time 0 in the order J_1, J_2, J_3, and J_4.



Part II – Long Form Question (**provide the shortest possible answer**)

1. **(25 points)** Consider the following Banker's algorithm problem. We have four processes A, B, C, and D. We have four resource types: R_1, R_2, R_3, and R_4. The availability vector for the four types of resources is given by [1, 2, 0, 1]. The MAX and HOLD matrices for the problem are shown below.

$$MAX = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 2 & 3 & 1 & 2 \\ 2 & 2 & 0 & 1 \\ 1 & 2 & 1 & 1 \end{bmatrix}, \qquad HOLD = \begin{bmatrix} y & 2 & \overset{1}{\cancel{0}} & 1 \\ y & 2 & 1 & 2 \\ 1 & x & 0 & 1 \\ 0 & x & 1 & 0 \end{bmatrix}$$

   a) What are the possible values we could have for *x, y* if we want the system denoted by the above matrices to be *very safe?*
   b) What are the possible values for *x, y* if we want the system to be *safe*?

$$Need = \begin{bmatrix} 2-y & 1 & 0 & 0 \\ 2-y & 1 & 0 & 0 \\ 1 & 2-x & 0 & 0 \\ 1 & 2-x & 0 & 1 \end{bmatrix} \qquad Avail = [1 \quad 2 \quad 0 \quad 1]$$

$\boxed{3}$

$\boxed{10}$

$$\left.\begin{array}{l} 2-y \leq 1, \quad y \geq 0, \quad y \leq 2 \equiv y = 1, 2 \\ 2-x \leq 2, \quad x \geq 0, \quad x \leq 2 \equiv x = 0, 1, 2 \end{array}\right\} \text{very safe.}$$

safe :

$\boxed{12}$

$x = 0, 1, 2$

$y = 0, 1, 2.$

2. **(25 points)** We have a process that is producing a data object that is consumed by another process. The two processes are using a shared memory region to exchange the data object. That is they are using a buffer in the shared memory region for the exchange.

   a. Show the pseudo code for data exchange if the buffer size is infinite. Your pseudo-code must show all semaphore initializations and operations.

   b. Modify the pseudo-code to accommodate a finite buffer size.

   c. Can your solution work with multiple producers?

   d. Can it work with multiple producers and multiple consumers at the same time?

$\boxed{2}$ Semaphore mutex = 1

$\boxed{2}$ Semaphore full = 0;

$\boxed{2}$ Semaphore empty = N;

```
producer () {
  while (1) {
    [3] wait(empty);
    [1] wait (mutex);
    [1] insert (item);
    [1] signal (mutex);
    [1] signal (full);
  }
}

consumer () {
  while (1) {
    [1] wait (full);
    [1] wait (mutex);
    [1] delete (item);
    [1] signal (mutex);
    [3] signal (empty);
  }
}
```

(b) need to complete with these additional lines

$\boxed{2.5}$ (c) yes, it can work with multiple producers

$\boxed{2.5}$ (d) yes, it can work with multiple producers and consumers.