EXAM I – Section 0201
CMSC-421

Any and all forms of cheating will result in a net score of 0.

Question 1 (*8 Points*)

Most round robin schedulers use a fixed size quantum. Discuss the pros and cons of a small quantum and a large quantum.

Question 2 (*12 Points*)

An OS supports two priority levels for processes, LO and HI. It implements the following scheduling algorithm:

a) If one or more HI processes are ready, schedule one at random;
b) Otherwise, pick at random a LO process and promote it to HI; then go to step a).

Assume that all processes use a single quantum of time and that the processes arrival rate is a newcomer at each quantum, with 50% probability of it being LO or HI.

1. (**2 points**) Suppose that when the system is activated only two processes A (LO) and B (HI) are present. What is the probability that A is completed after one quantum?
2. (**5 points**) After two?
3. (**5 points**) Can we guarantee that process A will be ever executed? Why or why not? Discuss.

Question (*15 Points*)

Two processes share the same critical section.

a) (**5 points**) What is the problem with the following code, assuming the syntax of the code below is correct and that we turn a blind eye to the *busy wait* condition?

```
repeat
  flag[i] = true;
  while flag[j] do nothing;
    .. critical section ..
  flag[i] = false;
    .. remainder section ..
until false;
```

b) (**5 points**) Another version is listed below. Do you see any problem?

```
repeat
  flag[i] = true;
  while flag[j]
    do begin
      flag[i] = false;
      delay random time
      flag[i] = true;
    end
  .. critical section ..
  flag[i] = false;
  .. remainder section ..
until false;
```

c) (**5 points**) Give a solution that eliminates problems seen in a), b) above. (Hint: Add another shared variable)

Question (*15 Points*)

1. (**8 points**) Describe the four necessary simultaneous conditions in a system for deadlock to occur?
2. (**7 points**) Consider two resource types A, B where there are 12 instances of each. Given the following snapshot of the system determine whether it is deadlocked and give a safe sequence if it is not deadlocked.

	Allocation		Request		Available	
	A	B	A	B	A	B
P0	2	0	2	4	4	5
P1	3	2	8	2		
P2	1	4	5	3		
P3	2	1	0	1		
P4	0	0	4	2		