NAME:____

CS 425/ECE 428 Distributed Systems **Fall 2015**
Mid-Term Exam 2 **Total points: 100**

1. (10 points) In the class, we discussed a mutual exclusion algorithm using a shared memory object V on which *test&set* and *reset* operations can be performed atomically. For your reference, the algorithm is shown in a slide at the top of page 1 of the attached handout.

Answer Yes or No in each part below. If you answer No, explain why.

(a) Will this algorithm achieve mutual exclusion if **linearizability** is guaranteed for any execution consisting of concurrent operations on the shared object V?

(b) Will this algorithm achieve mutual exclusion if **sequential consistenc**y is guaranteed for any execution consisting of concurrent operations on the shared object V? In this case, note that the execution is not guaranteed to be linearizable.

2. (10 points) Transactions T and U execute on a single server. In the table below, time proceeds from top to bottom, and the relative position of the operations indicate their relative order in time. Assume that a different lock is used for each variable w, x, y and z.

Assume that two-phase locking is used, and that each transaction acquires locks only when needed. For instance, in part (i) below, transaction T acquires lock for variable z right before it performs read(z). All locks held by a transaction are released when it performs closeTrasaction.

Determine whether the interleaving of operation as shown in parts (i) and (ii) below can occur under the assumption stated in each part (state YES or NO).

Notation: read (x) returns the value of variable x, and write(x,1) writes 1 to variable x.

(i) ***Assume that Exclusive locks are used with two phase locking***: ______________
If you answer NO, **delete** a minimum number of operations from the transactions so that the modified execution becomes possible while using exclusive locks. Assume that the relative order of remaining operations stays unchanged, even if some of the operations are deleted.

Transaction T	Transaction U
openTransaction	
	openTransaction
	b = read (y)
a = read (x)	
write (w,2)	
	c = read (x)
	write (z,1)
	write (x,5)
	closeTransaction
d = read (z)	
closeTransaction	

(ii) ***Assume that Read-Write locks are used with two phase locking***: ________________
(With read-write locking, a transaction acquires *read lock* when that is adequate. The transaction may promote a *read lock* to a *write lock* later if needed.)

If you answer NO, **change** a minimum number of **write** operations to **read** operations so that the modified execution becomes possible while using read-write locks.
See the execution on the next page.

Transaction T	Transaction U
openTransaction	
	openTransaction
a = read (x)	
e = read (z)	
	c = read (z)
write(x,5)	
	write (z,1)
	write(y,3)
	closeTransaction
write (y,2)	
closeTransaction	

3. (8 points) When using distance vector routing to maintain routes, is it possible for routing loops to form temporarily? Justify your answer.

4. (12 points)

(a) The table below shows the interleaving of the operations performed by transactions T, U and V. Suppose that optimistic concurrency control with *forward* validation is used. Validation is carried out when a transaction performs closeTransaction operation. When forward validation fails, **assume that the transaction performing the validation is immediately aborted**. Which of the transactions (if any) must be aborted?

Briefly explain your justification for each aborted transaction.

Transaction T	Transaction U	Transaction V
openTransaction		
	openTransaction	
		openTransaction
a = read (v)		
write (y,1)		
	e = read(w)	
		c = read (v)
		write (w,5)
		closeTransaction
write (z,5)		
	d = read (w)	
	write (w,5)	
	closeTransaction	
b = read (w)		
write (v,5)		
closeTransaction		

(b) When using timestamp ordering, suppose that two concurrent transactions T and U commit at time t1 and t2, respectively, where **t1 < t2**. Assume that there are no other transactions.

State true or false with a brief explanation: The above execution of T and U must necessarily be serially equivalent to an execution in which T executes in isolation first, and U executes in isolation after T commits.

5. (10 points) Assume the following:

- n processes with unique identifiers have access to a shared memory object V on which compare&swap operation can be performed atomically.
- Any concurrent execution of the n processes that includes compare&swap operations on V is guaranteed to be linearizable.

Using compare&swap operations on shared object V, present **pseudo-code for entry and exit sections** of a mutual exclusion algorithm. You may not use any other shared objects to implement your algorithm.

You may assume that V is initialized to any value that you need initially. Specify the initial value of V that you assume.

If you believe that such an algorithm does not exist, explain why.

6. (10 points) Consider Chord peer-to-peer network that uses 4-bit identifiers. Suppose that the network consists of nodes with identifiers 4, 8, 11 and 15.

(a) In the above network, for each key identifier below, determine the node that will store the key.

Key identifier 8 –

Key identifier 14 –

(b) If a node with identifier 10 is added to the network above, determine how many fingers at nodes 4 and 8 will need to be modified, if any, due to the addition of node 10. **Briefly explain your answer**.

Node 4:

Node 8:

(c) Which keys, if previously stored at one of the existing nodes in the system, will have to be moved to node 10, when node 10 is added? Specify the keys and node from which they will have to be moved.

7. (10 points)
(a) Draw a hypercube network consisting of 8 nodes, and label the nodes in binary.

(b) In a hypercube consisting of 2^6 nodes, consider two nodes that are at distance 4 from each other. What is the number of **pairwise edge disjoint shortest paths** (i.e., paths of length 4) between these two nodes? Two paths are said to be **pairwise edge disjoint**, if they do not contain any common edges (i.e., links).

Provide a justification for your answer.

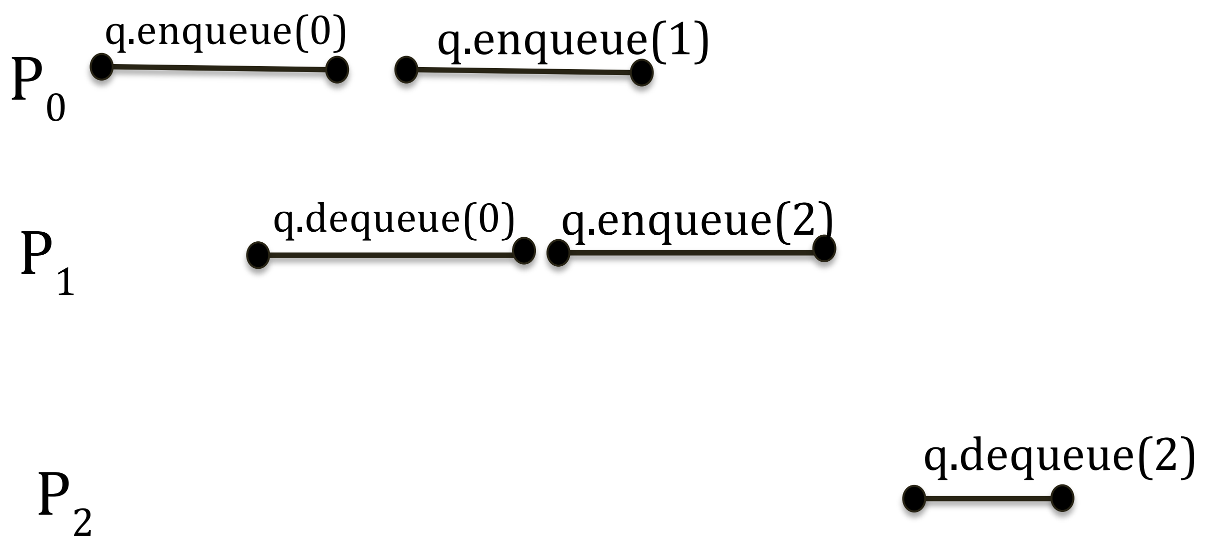
8. (10 points) The Paxos protocol discussed in the class is NOT designed to tolerate Byzantine failures. Consider an implementation of Paxos with 5 acceptors. Suppose that one of the acceptors suffers from Byzantine failure. Can this result in two different learners learning different values as being chosen?

If you answer yes, justify your answer using an example execution in which different learners learn different values as chosen. If you answer no, explain why.

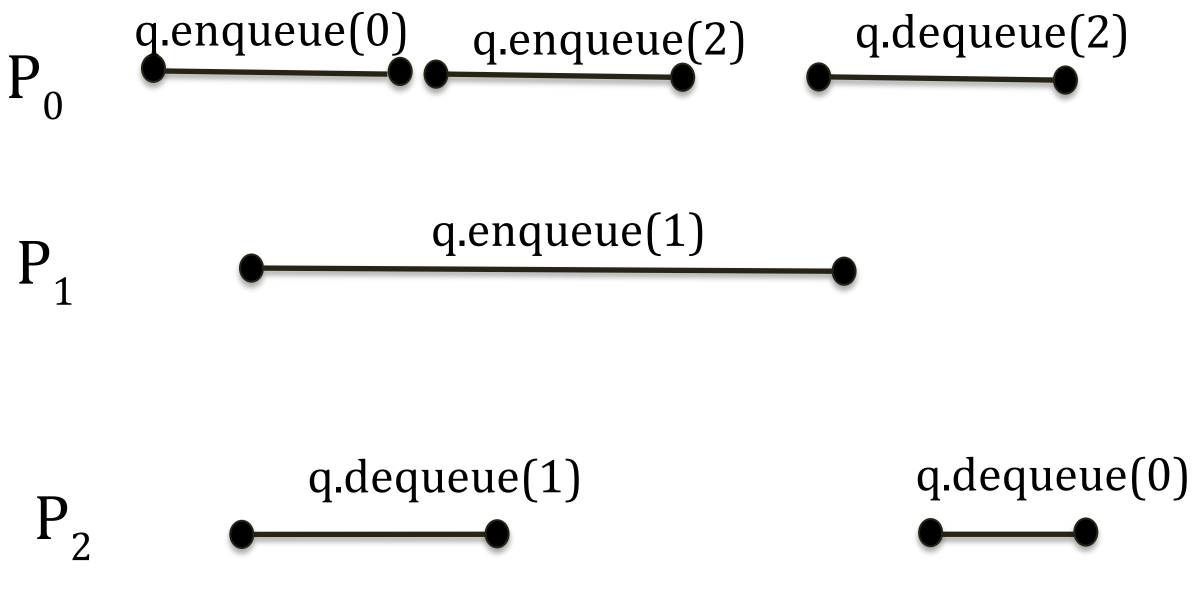
9. (10 points)
Consider the executions in parts (a) and (b) below for processors p_0, p_1 and p_2. Queue q is a concurrent queue. For each part, determine whether the execution is linearizable or not.

State YES or NO. If you answer YES, then mark the linearization points for all the operations (using triangles). If you answer NO, remove the smallest number of operations to make the execution linearizable.

(a)



(b)



10. (10 points) Design a mutual exclusion algorithm for a **message-passing** system consisting of n>2 processes named p0, p1, etc. The processes communicate with each other via messages (i.e., not shared memory).

The system is said to be in a quiescent state if none of the processes is in the entry or exit section of the mutual exclusion algorithm, or in the critical section (i.e., all processes are performing some computation unrelated to mutual exclusion or critical section).

When the system is in a quiescent state, if p0 decided that it needs to enter the critical section, then your algorithm must guarantee that p0 will be the first process to enter the critical section (even if other processes may concurrently attempt to enter the critical section).

Assume that the system is asynchronous, but no processes fail.

If you believe that such a mutual exclusion algorithm does not exist, explain why. Otherwise, describe your algorithm.