NATIONAL UNIVERSITY OF SINGAPORE

SCHOOL OF COMPUTING

SEMESTER 2 (2010/2011)

EXAMINATION FOR

CS4231: Parallel and Distributed Algorithms

April 2011 **Time Allowed: 2 Hours**

INSTRUCTIONS TO CANDIDATES

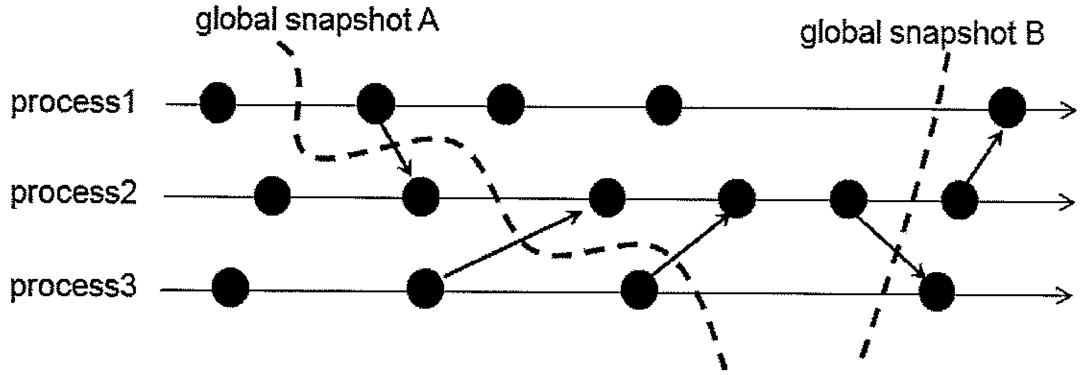
1. This examination paper contains **ELEVEN (11)** questions and comprises **NINE (9)** printed pages, including this page.
2. Answer **ALL** questions within the space in this booklet.
3. This is an **Open Book** examination.
4. Please write your Matriculation Number below.

MATRICULATION NO: ______________________________

This portion is for examiner's use only

Problem	Marks	Score
Problem 1, 2, 3	12	
Problem 4, 5	8	
Problem 6, 7	10	
Problem 8	6	
Problem 9	6	
Problem 10	8	
Problem 11	10	
Total	60	

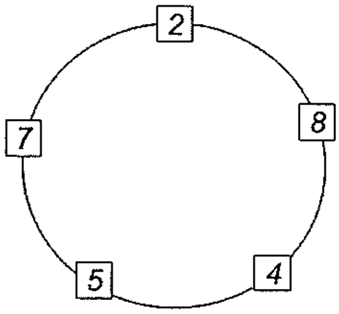
Problem 1. (4 marks)
Consider the following two global snapshots:



Is global snapshot A a consistent global snapshot? Your answer: _________ (yes or no)

Is global snapshot B a consistent global snapshot? Your answer: _________ (yes or no)

Problem 2. (4 marks)
Imagine that we run Chang-Roberts's lead election algorithm on the following ring where nodes only send messages clockwise:



What is the total number of messages incurred (assuming that the leader only needs to know that it is elected and does not need to announce itself to the other nodes)?

Your answer: ______________________ (just write the number, no need to explain)

Problem 3. (4 marks)
Consider the simple program "x = x+1" where x is a shared variable. Imagine that 5 processes execute this program concurrently (without mutual exclusion). Assume that the initial value of x is 0. After the 5 processes finish executing the program, what are the possible values for x?

Your answer: ___________________ (just write all the possibilities, no need to explain)

Problem 4. (4 marks)
Let X and Y be two semaphores. Let t and s be two shared integer variables. Alice wrote the following program for two processes:

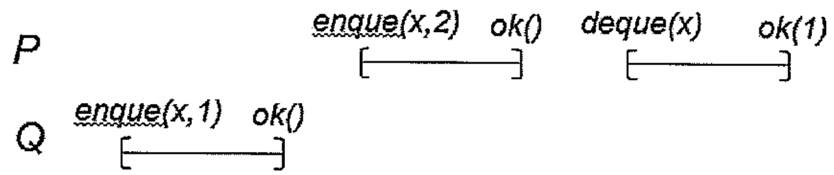
Code for P1	Code for P2
X.P(); Y.P(); t = s+1; X.V(); Y.V();	Y.P(); X.P(); s = t+1; Y.V(); X.V();

What problem will occur when running Alice's program? You need to point out the problem and explain why such problem may happen.

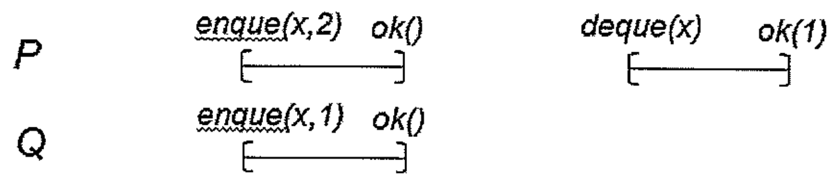
Write your answer below:

Problem 5. (4 marks)
Assume that the shared queue object, x, is initially empty. Consider the following histories:

History H1:



History H2:



Is history H1 sequentially consistent? Your answer: ________________ (yes or no)

Is history H2 sequentially consistent? Your answer: ________________ (yes or no)

Problem 6. (4 marks)
Consider the total order broadcast protocol that uses a coordinator. In this protocol, to broadcast a message, a process will send that message to the coordinator. The coordinator assigns a sequence number to the message, and then forwards the message and its sequence number to all processes. Messages are delivered according to their sequence number ordering. Does this protocol always satisfy causal order message delivery? If yes, provide a rigorous proof. If no, construct a counter example. Only indicating yes/no will not earn you any marks.

Problem 7. (6 marks)
Assume that you have implemented the logical clock algorithm. However, some application needs to use vector clock. Alice proposes designing a function that can convert a logical clock reading to a vector clock reading. Namely, the function takes (only) the local clock reading as input and outputs a vector clock value that satisfies the property which we need from standard vector clock readings. Do you think this is possible? If yes, indicate "yes" first, and then design such a function and prove the correctness. If no, indicate "no" first, and then concisely explain why this is not possible. Only indicating yes/no will not earn you any marks.

Problem 8. (6 marks)
Consider the self stabilizing spanning tree algorithm that we discussed in lecture. Assume that all the nodes execute their adjustment steps synchronously. Namely, for each adjustment step, all nodes (except the root) will first retrieve information from their neighbors simultaneously, and then all nodes (including the root) will adjust their local variable simultaneously. For any given topology and any node v in that topology, define depth(v) to be the length of the shortest path from node v to the root P in that topology.

You are asked to construct a fixed topology (with a root node P) such that by running the self stabilizing spanning tree algorithm, for any node v in the topology, v will **change** the value of its local "dist" variable exactly depth(v)+1 times. Furthermore, your topology should have at least one node v where depth(v) = 2. In your answer, you should first draw the topology and indicate the initial values of the "dist" variables at each node. Then you should clearly explain why it satisfies the property needed.

Write your original topology and the initial value of the "dist" variable on each node:

Explain clearly why this construction satisfies the property needed:

Problem 9. (6 marks)
Consider a java object X. Use Java monitor to implement the following 3 methods that can be invoked from inside the monitor of X:

```
void X.my_wait() { … }; // should have the same semantics as X.wait();
void X.my_notify() { … }; // should have the same semantics as X.notify();
void X.my_notifyAll() { … }; // should have the same semantics as X.notifyAll();
```

Your implementation is allowed to invoke the method wait() and the method notify() in Java, but is NOT allowed to invoke the method notifyAll() in Java.

Write your pseudo-code below and concisely explain why it is correct:

Problem 10. (8 marks)
Consider the coordinated attack problem where every node is reliable, every channel may drop messages, and the timing model is synchronous. A protocol solving the problem needs to satisfy all the following properties:

> **Termination**: All nodes eventually decide.
> **Agreement**: All nodes decide on the same value.
> **Validity**: If all nodes start with 0, decision should be 0. If all nodes start with 1 and no message is lost throughout the execution, decision should be 1. Otherwise nodes are allowed to decide on anything (but their decisions still need to agree).

In our lecture, we only studied this problem for two nodes P1 and P2. Consider the same problem for three nodes P1, P2, and P3, where each node can directly send messages to any other node (via an unreliable channel that may drop messages). Except for increasing the number of nodes from 2 to 3, all other conditions and requirements in the problem remain unchanged. Do you think the above coordinated attack problem for 3 nodes is possible to solve using a deterministic algorithm? If yes, indicate "yes" first, and then design a deterministic protocol to solve the problem and **rigorously** prove the correctness of the protocol. If no, indicate "no" first, and then **rigorously** prove such impossibility. Only indicating yes/no will not earn you any marks.

Problem 11. (10 marks)
Consider the following *grouping problem* whose goal, intuitively, is to group the nodes in a distributed system into no more than 3 groups. Formally, there are n nodes and each node can directly send messages to any other node. The communication channels are reliable, and up to f nodes may experience crash failures. It is known that $n > 2f > 100$. A node is nonfaulty if it does not crash during the protocol execution. The timing model is synchronous. Each process has an initial integer input between 1 and 9 (inclusive). Intuitively, you can view the input as the "initial group number" proposed by the process. You are asked to design a deterministic protocol so that

i) every nonfaulty node is eventually assigned a "final group number", and
ii) any "final group number" assigned (to any node) must be the initial input of some node, and
iii) among all the nodes, there are at most 3 different "final group numbers".

Intuitively, by using this protocol, we will be able to group all the nodes into no more than 3 groups, according to their "final group numbers".

You should do your best to minimize the number of rounds needed by your protocol in the worst case. Your mark will partly depend on the efficiency (i.e., the number of rounds needed) of your protocol. Your answer should consist of 4 parts:

i) indicate clearly the total number of rounds needed by your protocol,
ii) give an intuitive description of the idea behind your protocol,
iii) give concise pseudo-code for the protocol, and
iv) rigorously prove the correctness and performance of your protocol.

Write your answer below:

(this page is intentionally left blank for you to write your answer for Problem 11)

END OF EXAM PAPER