**Question 1:**

This question relates to quorum replication (of R/W registers).

a) The implementers of a certain key-value datastore decided to skip line 4 of the client read protocol that we have seen in class, claiming that in the workloads they envision, the atomic register semantics would be preserved even without it, so it is not worth the extra latency. Explain what properties should such a workload exhibit in order for their claim to be correct? Explain your answer.

b) Suppose the timestamp in the client write protocol is obtained from an atomic clock (plus the process id to break symmetry) – in this case line “A” becomes setting the timestamp to the output of the atomic clock and lines 1&2 (in the following slide) are not needed. Is line 4 still needed in the client read protocol to ensure atomic register semantics? Explain your answer.

**Question 2:**

The following variant of Mostefaoui\&Raynal has been proposed:

Code for i

1: **init:** ri ← 0, esti ← vi
2: **while** didn’t decide **do**
3: \[c ← Ω; estc ← nil; ri ← ri + 1\]
4: \[\text{if } i = c \text{ then} \]
5: \[esti ← esti\]
6: \[\text{else} \]
7: \[\text{wait until } <EST,ri,v> \text{ is received from } c \text{ or } c \notin Ω \]
8: \[\text{if } <EST,ri,v> \text{ is received from } c \text{ then} \]
9: \[estc ← v\]
10: \[\text{end if} \]
11: \[\text{end if} \]
12: broadcast(<EST,ri,estc>)
13: \[\text{wait until } <EST,r’i,est’c> \text{ messages was received from } n - f \text{ nodes} \]
14: \[reci ← \{est’c | <EST,r’i,est’c> \text{ was received}\}\]
15: \[\text{if } reci = \{v’\} \text{ then} \]
16: \[\text{decide } v’ \text{ and broadcast(<DECIDE,v’>)} \]
17: \[\text{else if } reci = \{v’,nil\} \text{ then} \]
18: \[esti ← v’\]
19: \[\text{end if} \]
20: **end while**
21:  
22: **Upon receiving** (< DECIDE, v 0 > from q **do**
23: \[\text{decide } v 0 \text{ and broadcast(< DECIDE, v 0 >)}\]

Does it solve the consensus problem? Explain your answer – in case your answer is positive, NO NEED for a formal proof.
**Question 3:**

In this question we will prove the following lemma regarding Paxos.

**Lemma:** Suppose a quorum of acceptors have sent \((\text{ACCEPTED}, r, \text{ACK})\) messages for some value \(v\) in some round \(r\). Then no leader with a round number \(r_1 > r\) would send an \((\text{ACCEPT}, r_1, v_1)\) message with \(v_1 \neq v\).

**Proof:** Assume, by way of contradiction, that the theorem does not hold.

**Complete the proof from here.**

Hint, think of a minimality argument regarding a leader that sends \((\text{ACCEPT}, r_1, v_1)\) with a value \(v_1 \neq v\).

**Question 4:**

a) Explain intuitively the following claim: “\(S\) based consensus protocols require all participants to agree on the entire membership while \(\Omega\) based consensus protocols can solve consensus with weaker knowledge guarantees about the membership”.

b) Can Paxos be used as is to solve the Non-Blocking Atomic Commit problem, meaning that each process invokes Paxos with “Abort” or “Commit” and decides on the output of Paxos? Explain your answer.

**Submission instructions:**

You should solve this exercise alone — submissions are individual. Solutions must be submitted through the course web site – either printed or a high-resolution scan of handwriting. Solutions must be written in Hebrew unless you get an authorization from Prof. Friedman to submit in English.

Try to be brief. If your answer is very lengthy, it could be a sign that it is wrong.

The submission date is Sunday 02/12/2018 before 12:00 noon.

**Good luck!**