# Assignment 1

Question 1. Read week1.pdf (that I wrote earlier) and give me comments if you have any (e.g., typos you found, or suggestions for improvement)

Answer 1. No improvisation required for week 1.pdf. I found it quite understandable and simple.

Question 2. Describe an application problem that is interesting to you and that requires a distributed algorithm to solve.
First, describe what the problem is---what needs to be computed or achieved but not how to compute it. That is, describe what is given as input and what is asked as output, including any restrictions on the input and how the output is related to the input, but not how to go from the input to the output. Then, describe what the application domain is or why the problem is interesting to you, if this is not already obvious. Finally, describe why this problem requires a distributed algorithm to solve.

Answer 2. The most interesting distributed algorithms problem for me is its usage in banking transactions.

- Input : Different transactions of debit and credit within a single person account from various sources for example net banking, putting cash directly in ATM.
- Output : Timestamping the debit and credit transactions in proper order from different sources so that the money is neither more or less than what it should be.

The problem is interesting to me because the number of messages passed and the timestamping used in transferring messages cannot be risked by even 1 minute or it can lead to total chaos in bank account.

Question 3. How to implement message passing for concurrent systems using shared memory?

Answer 3. The messages passed from one process can pass through the shared memory. The messages can have 3 things in them :

- The task or signal of what to do
- The timestamp at which it was generated
- The task for which it was pointed

There could be a global element which is accessible for all the processes. Every process can take the required message from the shared memory.

Question 4. What is the maximum value of a Lamport clock in a system at any time?

Answer 4. The maximum value of lamport logical clock would be the time at which the last event within the process took place.This would even include the last message sent from the process or the last message received from any other process.

Question 5. In Lamport's logical clock paper, the implementation rule 2(b) says "Upon receiving a message m, process Pi sets Ci greater than or equal to its present value and greater than Tm. Should "greater than or equal" be changed to "greater" instead? Why?

Answer 5. The message will have a particular timestamp Tm on it. When the other process Pj will receive the message the clock Cj for Pj should be only set “greater than” Tm as sending a message by a process is also an event and within every event the process clock advances as discussed earlier in the

paper. Therefore, if a process Pi sends a message at Ci, the Process Pj will receive the same message at Cj such that Cj>Ci as message receiving event in Pj and message sending event in Pi will be separated by advancing clock ticks in between. Hence, the time Cj and Ci can never be equal.