You are to work on the following questions *alone*. Do not discuss these questions with anyone. Typeset your answers and submit as a PDF.

1. (10 points) **Paxos Acceptor States**

   Consider a deployment of single-instance Paxos with three acceptors. Decide whether each of the following is a valid state of the three acceptors. If the state is not valid, explain why in one sentence. (Hint: A state is valid if there is some sequence of message deliveries and message drops and node failures that leads to the state, assuming a correct implementation of proposers and acceptors.)

   For each part, we give you the highest accepted proposal at all three acceptors \((A_1, A_2, A_3)\) at a single instance in time. Each acceptor’s highest accepted proposal is either in the form \((n, v)\) where \(n\) is the proposal number (à la Paxos Made Simple) and \(v\) is a value or \(⊥\) which indicates that the acceptor has not accepted any proposal.

   (a) \(A_1: ⊥, A_2: ⊥, A_3: ⊥\)
   (b) \(A_1: (1, x), A_2: (2, y), A_3: ⊥\)
   (c) \(A_1: (2, x), A_2: (2, y), A_3: ⊥\)
   (d) \(A_1: (1, x), A_2: (2, y), A_3: (3, z)\)
   (e) \(A_1: (1, x), A_2: (2, x), A_3: (3, x)\)

2. (10 points) **Acceptor States in a Larger System**

   Consider a deployment with five acceptors. Is the following state valid? If it is valid, describe an execution that results in this state. If it is not valid, explain why.

   \(A_1: (20, x), A_2: ⊥, A_3: (22, y), A_4: (20, x), A_5: (18, x)\)

3. (10 points) **A Dubious Execution**

   Consider another Paxos deployment with acceptors \(A_1, A_2, A_3\), proposers \(P_1, P_2\), and a distinguished learner \(L\). According to the Paxos paper, a value is chosen when a majority of acceptors accept a proposal with that value, and only a single value is chosen. How does Paxos ensure that the following sequence of events cannot happen? What actually happens, and which value is ultimately chosen?

   - \(P_1\) prepares proposal number 1, and gets responses from \(A_1, A_2, A_3\).
   - \(P_1\) sends \((1, x)\) to \(A_1\) and \(A_3\) and gets responses from both. However, \(P_1\)’s proposal to \(A_2\) was dropped. Because a majority accepted, \(P_1\) informs \(L\) that \(x\) has been chosen. \(P_1\) then crashes.
   - \(P_2\) prepares proposal number 2, and gets responses from \(A_2\) and \(A_3\).
   - \(P_2\) sends \((2, y)\) messages to \(A_2\) and \(A_3\) get responses from both, so \(P_2\) informs \(L\) that \(y\) has been chosen.
4. (10 points) **Paxos Liveness**

In the absence of a distinguished proposer, it is possible for Paxos to fail to make progress even if no messages are dropped and no nodes fail. Briefly describe how this can happen in a system with two proposers and three acceptors. Be specific about which messages are sent and in what order they are delivered.

5. (10 points) **Alternate Paxos Implementation**

The *Paxos Made Simple* paper has the following definition in page 3.

A value is chosen when a single proposal with that value has been accepted by a majority of the acceptors.

Consider pursuing an alternate implementation based on the following definition.

A value is chosen when proposals with that value have been accepted by a majority of the acceptors.

Would the resulting implementation be correct? Justify your answer in a few sentences either with an informal proof or a scenario where this implementation would violate safety.