1. Consider a distributed system where two clients generate the following requests to read and write variables from a distributed shared memory. Each item below lists the request issued by the client, the time at which the request was issued by the client to the system (as per a global wall clock), and the time at which a positive acknowledgement response was received at the client.

- Client 0 issues two writes $x = 0$ and $y = 0$ at time $t = 0$, gets a response at $t = 0.5$.
- Client 1 issues a write $x = 1$ at $t = 1$, and gets a response at $t = 1.5$.
- Client 2 issues a write $y = 2$ at $t = 2$, and gets a response at $t = 2.5$.
- Client 3 issues reads for $x$ and $y$ at time $t = 3$.
- Client 4 issues writes $x = 4$ and $y = 4$ at time $t = 2.99$ and gets a response back sometime after $t = 3$.

State all possible values of $x$ and $y$ that can be obtained during the read at client 3, when the distributed system guarantees the following consistency models.

(a) Strict consistency.
(b) Atomic consistency / linearizability.
(c) Sequential consistency.
(d) Eventual consistency.

2. Consider Figure 7 in the Raft paper. For each follower shown, explain a scenario where the follower would have ended up the situation shown in the figure. Also, based on the restriction of leader election, which of the followers (a) through (f) would have voted for the new leader?
3. Consider a set of 5 replicas N1–N5 maintaining a replicated log by running the RAFT consensus protocol. Shown below are the logs at each of the 5 nodes. Entries $a, b, c$ are from term 1, entries $p, q, r$ are from term 2, and entries $x, y, z$ are from term 3. The nodes are about to elect a leader for their fourth term.

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>p</th>
<th>q</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>N1</td>
<td>a</td>
<td>b</td>
<td>p</td>
<td>q</td>
<td>r</td>
</tr>
<tr>
<td>N2</td>
<td>a</td>
<td>b</td>
<td>c</td>
<td></td>
<td></td>
</tr>
<tr>
<td>N3</td>
<td>a</td>
<td>b</td>
<td>p</td>
<td>q</td>
<td>x</td>
</tr>
<tr>
<td>N4</td>
<td>a</td>
<td>b</td>
<td>p</td>
<td>q</td>
<td></td>
</tr>
<tr>
<td>N5</td>
<td>a</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(a) Which of the five nodes can be elected as leader in term 4? For each node that you think can be elected leader, list the set of followers who will vote for it, to demonstrate how the node can achieve a majority vote.

(b) Can entry $c$ in the log be committed in term 4 (without the client retrying the request again)? Answer yes/no and justify your answer. If yes, describe the scenario under which this can happen. If you answer no, explain why not.

(c) Repeat part (b) for entry $r$.

(d) Repeat part (b) for entry $z$.

4. Consider an implementation of a distributed key-value store, where multiple clients issue requests to get/put key-value pairs. Shown below are multiple requests issued by a client, along with the time the request was issued and the time that the completion response was received.

- put $(k, v_1)$ at time $t = 1$, completed at $t = 1.5$.
- put $(k, v_2)$ at time $t = 2$, completed at $t = 2.5$.
- put $(k, v_3)$ at time $t = 3$.

You are not told when the response for the third put request was received. Now, the client issues get $(k)$ to the key-value store at $t = 3.5$.

(a) If the key-value store provides atomic consistency (linearizability), what is the set of possible values that can be received by the client? You must list all feasible values under all scenarios, and there is no need to explain your answer.

(b) Repeat part (a) if the key-value store provides sequential consistency.

(c) Now, you are told that the client has issued the get $(k)$ request at $t = 3.5$ to two different replicas simultaneously, and obtained a value of $v_1$ from one of the replicas. What is the set of possible values that can be received from the other replica, if the key-value store provides a sequential consistency guarantee?
(d) Repeat part (c) above if the key-value store only promises eventual consistency.

5. Consider three nodes A, B, C in a distributed system. The nodes wish to run a distributed transaction using the two-phase commit protocol. Node C acts as the transaction coordinator, and starts a new transaction. Describe what happens in the following failure scenarios.

(a) The nodes finish the first phase of the transaction. Node A replied yes to the prepare message in the first phase. Subsequently, A does not receive the commit or abort message of the next phase from the coordinator, and times out while waiting for this message. What does A do in this situation, as per the 2 phase commit protocol studied in class?

(b) The coordinator C sends a prepare message in the first phase to nodes A and B. A replies yes to C, but B fails before it can respond. C times out while waiting for a reply from B. What does C do in this situation?

6. Consider a distributed key-value store with multiple nodes. The set of keys is partitioned among the nodes such that a key is stored at only a subset (of size greater than one) of nodes in the system. When a get/put request for a key arrives to the system, the request is directed to all the nodes responsible for the key. Suggest one method of partitioning keys to nodes, and describe how one can compute the nodes responsible for a given key using your method.