1. **Asynchronous Consensus** (10 points)

   Consider an asynchronous system with 6 processes, at most one of which can fail. A failure consists of a process crash (i.e., the process stops executing actions). Channels are reliable so every message that is sent is guaranteed to arrive after some finite (though unbounded) delay.

   Each process begins with an initial value consisting of a single bit (i.e., 0 or 1). Consider the following algorithm for solving consensus:

   Every process sends its initial bit to all processes (including itself) and waits for 5 messages to arrive. Every process decides on the majority of the five bits received.

   Is this algorithm correct? If so, prove it is correct. If not, construct a counter-example. That is, construct a scenario in which the algorithm does not satisfy all three of the required properties of consensus.

2. **Binary Consensus** (4 points)

   The validity requirement is usually stated as:

   If every process has the same initial value, (correct) processes decide on that value.

   Consider the following alternative formulation:

   Every (correct) process’s decision value is the initial value of some process.

   For the specific case of binary consensus (i.e., initial values taken from a set of 2 values), show that these two formulations for validity are equivalent. For the case where initial values are taken from a set of more than 2 values, show that these two formulations are not equivalent.

3. **Lower Bounds** (10 points)

   In class we saw a synchronous algorithm for achieving consensus in the presence of crash faults that required \( f + 1 \) rounds, independent of the number of processes \( n \). This algorithm is optimal in the sense that no correct algorithm can always terminate in less than \( f + 1 \). In particular cases, however, it may be possible to terminate in fewer rounds.

   Modify the algorithm given in class so that it requires only \( f \) rounds in the case where \( f = n - 1 \).
4. Synchronous Consensus (10 points)

**Part I.** Consensus in the presence of Byzantine faults is guaranteed only if \( N > 3f \). Give an example of an execution of the Byzantine Agreement algorithm (without authentication) with 6 processes, 2 of which are faulty, in which consensus is *not* achieved.

**Part II.** Multiple rounds are required to achieve consensus in the presence of multiple Byzantine faults, even if authenticated messages are available. Consider modifying the Byzantine agreement algorithm (with authentication) so that it uses only 2 rounds. Give an example of an execution of this modified algorithm in which consensus is *not* achieved, despite \( N > 3f \).