Distributed Algorithms - Homework 2

1. The question relates to Peterson’s 1st algorithm to find a leader in a uni-
directional ring.
   Give a proof of correctness.
   What is the worst case (longest-chain) time complexity of the algorithm?
   Give an example that achieves this bound.

2. The question relates to Peterson’s 2nd algorithm to find a leader in a uni-
directional ring.
   • Prove that the algorithm always determines exactly one processor as
     a leader.
   • Show that this leader is not always the processor with the maximal
     identity. Give an example with as few processors as possible where
     this happens.
   • Modify the algorithm so that it will determine the processor with
     largest identity as a leader.
     Note: It is not allowed to first isolate a single processor, and then have
     it send a message around the ring to identify the maximal identity.
   • Prove the correctness of your algorithm.

3. The question relates to the Burns-Pachl algorithm for self-stabilization.
   Prove or disprove:
   The algorithm works correctly when the following conditions are removed
   from Rule A (one at a time):
   • a) $L_i \neq 0$
   • b) $T_{i-1} = 0$
   • c) $T_{i-1} \neq L_i - L_{i-1}$
   • d) $T_{i-1} < T_i$
   Note: Whenever you disprove, give a counter example (an execution) which
   is as short as possible.

4. The question relates to the Burns-Pachl algorithm for self-stabilization.
   Prove that the algorithm for self-stabilization does not work correctly on a
   ring with a prime number of processors if the scheduler is distributed. Give
   a relaxed assumption (as weak as possible) on the scheduler, so that the
   algorithm will not fail.
Note: a distributed scheduler chooses at each stage a subset of the processors. If this subset will always be of size one (namely, the scheduler is centralized), then we know that the algorithm stabilizes. The question is to identify the strongest scheduler which will still stabilize.