Some of the answers to these questions can be found using the lecture slides, the recommended textbooks or other sources (question marked with *). Some questions may have more than one possible answer, or be more or less open for discussion. Note that no answers (solutions) will be given to these questions, but if help is needed the assistants will be available to answer questions. The concepts marked with yellow are important and should be fully understood.

Distributed Systems

1. Discuss advantages and challenges resulting from distributed computing.
2. Explain and discuss the following concepts:
   (a) Networked Operating Systems (NOS) vs. Distributed Operating Systems (DOS)
   (b) Explain the various distribution transparency aspects presented to you during the lecture.
   (c) Define the scalability of a distributed system.
   (d) What is the rationale behind the middleware (MW) concept? Which gap in distributed systems (DS) are MWs trying to close?
3. Explain the need of having an elected leader in a DS.
4. Two-Phase Commit (2PC) Protocol - Basics
   (a) Explain how 2PC works in the fault free case. 2PC consists of four steps which are divided in two phases. Put your explanation in the context of these steps and phases. Sketch a timeline diagram involving 3 participants and a coordinator.
   (b) Sketch two state transition diagrams, one for the coordinator and another one for participants in 2PC. Diagrams should include the 2PC messages that induce state transitions.
   (c) Why can blocking operations in distributed algorithms be problematic? Reason based on 2PC and describe states where participants or coordinator could block. What is the basic mechanism used to overcome this issue?
5. Two-Phase Commit (2PC) Protocol - Fault Tolerance
   (a) Participants may talk to each other to find out about the state of other nodes. For example, when a participant p has reached the *commit* state because the coordinator has sent the *commit* message to p right before crashing and could not send the message to another participant q. What could q decide based on knowing p's state?

(b) What happens, if a participant is in the *ready* state, does not receive the coordinator response for a very long time and all other participants are in the *ready* state as well?

(c) It is important that nodes store their state in a log to read it back after crash recovery. What happens, if a participant crashes in *ready* state and then recovers; should it commit or abort?

(d) Which states are being logged by the coordinator?