1. What is significant for a client server architecture?
A the client is the active part
B servers have more execution power
C several clients but only one server
D the server is the active part
2. What would we call a system where one node is always reacting on requests and other nodes only communicate with this node?
A an asynchronous system
B a client server system
C a peer-to-peer system
D a synchronous system
3. What is significant for a peer-to-peer architecture?

A no single node may initiate an operation
B nodes are structured in a hierarchy of client and servers
C all nodes can communicate directly with all other nodes in the network1
D all nodes are active and can be the initiator of operations
4. What do we know in a synchronous system?
A that all operations will take equal amount of time
B exactly how long time it takes to deliver a message
C that all messages will be delivered
D the upper bound of the time to perform an operation
5. What is significant for an asynchronous system?
A that all operations will take equal amount of time
B that our internal clock has a bounded drift
C that the time to perform an operation does not have an upper bound
D that messages might not be delivered
6. What do we call a system where the maximum times for operations and message delivery are known?
A a fault tolerant system
B a real time system
C an asynchronous system
D a synchronous system

7. What is true if A happened real-time before B?
A B is a consequence of A
B A and B occurred in the same process
C A could have happened before B
D B could have happened before A
8. What is true for events A and B?
A if A caused B then A happened before B 13
B if A happened before B then A caused B
C if A occurred in real-time before B then A happened before B
D if A occurred in real-time before B then A caused B
9. What can we know if we use Lamport clocks?
A if L(a) < L(b) then a happened before b
B if L(a) < L(b) then a could have caused b
C if L(a) < L(b) then a must have caused b
D if L(a) < L(b) then a occured in real-time before b
10. What can we know if we use Lamport clocks?
A if, and only if, L(a) < L(b) then a happened before b
B if, but not only if, L(a) < L(b) then a happened before b
C if, but not only if, a happened before b then L(a) < L(b)
D if L(a) = L(b) then a = b
11. What can we conclude looking at the Lamport clock timestamps of two events?
A if L(a) < L(b) then a happened after b
B if L(a) > L(b) then b happened before a
C if L(b) = L(a) then we cannot conclude anything about a and b
D if L(a) ≮ L(b) then a did not happened before b 14
12. What is the most that we know if we use vector clocks?
A V (a) < V (b) if and only if a happened before b
B if V (a) < V (b) then a happened before b
C if a happened before b then V (a) < V (b)
D if V (a) = V (b) then a and b are unordered
13. What is the definition of a consistent cut?
A all events in the cut are strictly ordered in a happened-before order
B if e happened-before f and e is in the cut then f is in the cut
C if e and f are in the cut then e and f have a causal order
D if e is in the cut and f happened-before e then f is in the cut
14. What is the definition of a stable global state predicate?
A if a system enters a state where the predicate holds true it will remain true in all future states
B the predicate holds true in all consistent global states
C the predicate will eventually hold true in all linearizations
D the predicate will never hold true in any consistent state reachable from the original state
15. What is the definition of a unstable global state predicate?
A if a system enters a state where the predicate holds true it will remain true in all future states
B the predicate holds true in all consistent global states
C the predicate could hold true in a state but then be false in future states
D the predicate will never hold true in any consistent state reachable from the original state

16. Give an example of a stable global state predicate.
A none of the alternatives are stable
B deadlock
C the cargo door is open at 10.000 meters
D A is waiting on a message from B and B is waiting for a message from A

17.**What is the difference between a Lamport clock and a vector clock?**
A nothing, two names for the same thing
B only difference is that Lamport clocks are more efficient since they require smaller messages
C only the vector clock gives a complete description of the ”happened before order”
D only the Lamport clock gives a complete description of the ”happened before order”
18 When is it problematic to use vectors clocks?
A when we have a dynamic set of processes 25
B if nodes are not synchronized using for example NTP
C if we have an asynchronous system
D when we do not have an elected leader process
19 An alternative way of implementing a vector clock would be to keep a set of the highest counters seen from each process (including own), send it along with any message, update own counter and merging the own set and received set when a message is received. This would have the following advantage:
A new processes can easily be added
B the set is easier to represent
C nothing, it is identical to vector clocks
D events would become totally ordered
20. If events in a given set are to be ordered in a total order that respect the happen-before order what is the advantage of using vector clocks?
A only vector clocks will give us the happen-before order
B only Lamport clocks will give us the happen-before order
C if two events are unordered only vector clocks can order them
D no advantage at all
21. What do we know if we record a snapshot using the algorithm by Chandy and Lamport?
A the execution passed through the state described by the snapshot
B a non-stable predicate that is true for the snapshot has also been true during the execution
C if the algorithm terminates then the execution will also terminate
D there is a linearization from the original state to the final state that passes through the

22 Assuming that we collect all state transitions of nodes and have them tagged with vector clocks what can we then do?
A for any unstable predicates determine if it possibly was true during the execution
B for any unstable predicates determine if it was true during the execution
C determine all stable, but no unstable, predicates
D determine termination but not deadlock
23. How can we detect that a non-stable predicate definitely was true during an execution?
A let each node evaluate the predicate based on local state
B generate all consistent runs and show that the predicate is true at one point in all
C collect a snap-shot and examine if the predicate is true
D since the predicate is non-stable it cannot be determined