Where an algorithm is asked for, you may write in any suitable pseudocode. Correct syntax for any computer language is not expected.

Answer 3 questions.

1.

a) Explain what is meant by the terms *local delay*, *remote delay, sequential processes,* and *synchronous communication.* [8]

b) According to Liskov, Herlihy, and Gilbert

> 'To provide adequate expressive power, a language for distributed programs should abandon either synchronous communication primitives or the static process structure'

To what extent do you agree with this statement? In answering this you might like to consider what the alternatives are, which of them you consider to be the best, and in what circumstances. [25]

2.

a) Explain what is meant by a *schedule* and by *two phase locking*. [4]

b) Serialisability in a distributed system cannot simply be ensured by ensuring that the schedules at each site are serialisable. Why not? [6]

c) Prove that two phase locking ensures serialisability in a distributed transactional system. [11]

d) Which concurrency control mechanism would be the most appropriate in each of the following scenarios and why? If you cannot decide on a clear favourite, then explain what the advantages and disadvantages of each possible approach are.

i) A large number of transactions execute, but very few of them are in conflict in terms of the data they access. [4]

ii) A large number of short transactions execute, but many of them abort. [4]

iii) There are a few long running transactions and many short transactions. [4]

[TURN OVER]

3.

a) Explain what is meant by *weighted voting*. [4]

b) The generalised quorum consensus approach to replication is more flexible than weighted voting. Does this mean that one should necessarily use generalised quorum consensus in preference to weighted voting? Justify your answer. [7]

c) The Borings bank uses a transactional system to record the dealings of its traders on the foreign exchange money market. Logging is used both to enable effective recovery and as an audit trail, so that the company and its auditors can, at any moment, see precisely what changes have been made, to what accounts, and by whom. To minimise the chances of the audit trail being physically destroyed by a villain, it is replicated several times in different places throughout the country.

   What replication scheme would you consider to be most appropriate for this application and why? [11]

d) The company currently has two replicas and these are held on reliable machines (down 1 hour per week). Unfortunately, such machines are expensive. They are considering using a cheaper alternative (down 1 hour per day) which work out at 25% of the cost of the reliable machines or a very cheap solution (down 1 hour in 5) which are 10% of the cost of the reliable machines.

   i) Using these figures and the scheme you proposed in (c), what is the availability of the audit trail data in the current setup.

   ii) What would be the most cost effective solution for the company if they wish to retain or improve on the degree of availability which is currently achieved. [11]

4.

a) Explain the purpose of Public Key Certificates and suggest a suitable structure for them giving reasons for your choice of fields. [13]

b) A company of 10,000 employees wishes to use encryption techniques to secure its data. Not only does it wish to protect against interlopers, it is also concerned that disclosure of information through loss of keys should be limited and that if employees are away sick that critical information should not be inaccessible to other appropriate employees.

   Suggest security procedures and architectures which will address the problems you identify in this scenario. Discuss the strengths and weaknesses of your chosen solution. [20]

[CONTINUED]

5. You wish to debug a distributed program. In order to do this, you need to obtain a global snapshot of the state of the application, consisting of the states of each of the processes involved. Since you cannot actually ensure that all the snapshots occur at precisely the same time, you decide to adopt an approach in which, when any event of interest occurs, a message is sent to your machine and its output appears on your terminal window. You would like it if the messages were displayed in a meaningful order, rather than just in whatever order they happened to arrive at your machine, and you would like to be able to determine what the state of the application is at any point in time.

   One possible way of doing this would be to have synchronised clocks and to timestamp every event, then print them out in timestamp order. However, you do *not* have synchronised clocks in your system.

   a) What is a Lamport logical clock, and how is one implemented? [6]

   b) Assume each process keeps a vector of logical clock values which contains one slot for each of the processes in the system. How might the vector be updated? Does this extra information have any value? If so, what; if not, why not? [8]

   c) Define the notion of a global consistent state for the application, which must be assembled from the messages sent to the debugger/display process. Justify your definition. [8]

   d) Assuming that message delivery is not ordered, how can one tell if a reconstructed global state is consistent or not? [11]

[END OF PAPER]