Answer question 1 and any other two questions.

1. Answer the following four parts.

   a. Explain in no more than 100 words the difference between monitors and semaphores. Pay particular attention to why monitors lead to better designs of concurrent programs.

   [8 marks]

   b. In the course we discussed how monitors could be used to implement semaphores. In practice the need to construct a monitor using semaphores arises more frequently. For the following FSP specification sketch how you would implement a monitor using semaphores in an object-oriented programming language of your choice.

```
CARPARKCONTROL( N=4) = SPACES[ N],
SPACES[ i: 0.. N] =
  ( when( i> 0) arrive -> SPACES[ i- 1]
  | when( i< N) depart-> SPACES[ i+ 1]).

ARRIVALS = (arrive-> ARRIVALS).

DEPARTURES = (depart-> DEPARTURES).

||CARPARK = (DEPARTURES || ARRIVALS || CARPARKCONTROL).
```

   Assume that there is a class `Semaphore` with the two operations `up` and `down` available.

   [9 marks]

c. A World-Wide-Web (WWW) navigation session is defined as the activity of browsing through a sequence of WWW pages by the process of following links. Such a sequence of WWW pages is called a *trail*. Let us assume that you have available a function called, *useful*, which given a set of keywords and a trail, returns a number between zero and one indicating how "good" the trail is.

Give the pseudo-code of a non-deterministic algorithm that would automatically find a trail whose usefulness is greater than a half. The input to your algorithm is a starting node for the navigation, a set of keywords and the required length of the trail to be found. Briefly explain the main idea behind your algorithm.

[9 marks]

d. Give the pseudo-code of a *probabilistic* solution to the dining philosophers problem. In what sense is this solution preferable to the solution using semaphores ?

[8 marks]

[Total 34 marks]

2. a. Write a maximum of 50 words on each of the following:

   i. The difference between Processes and Threads

   ii. Critical sections

   iii. Critical regions

   iv. Safety properties

   v. Liveness

   vi. Starvation

   vii. Deadlock

   viii. Livelock

[16 marks]

b. Use FSP to specify a system of traffic lights for a junction of two roads. Define the system in such a way that there is one traffic light for each of the four directions. Assume that each traffic light is controlled by a separate process. Neglect special lights for pedestrians and cyclists.

[9 marks]

c. Specify safety properties in FSP for the traffic light system above so it can be proven, by way of reachability analysis, that the traffic lights do not cause car accidents.

[8 marks]

[Total 33 marks]

3. a. Explain the interleaved model of concurrency using the Java thread model. Elaborate on the Java thread life cycle and discuss how the life cycle supports interleaved concurrency.

[15 marks]

b. The following is a FSP specification of a one way bridge:

```
const BRIDGE_CAPACITY=5
range T=0..BRIDGE_CAPACITY
DIRECTION=DIRECTION[0],
DIRECTION[i:T]=(
   when (i<BRIDGE_CAPACITY) enter->DIRECTION[i+1]
 | when (i>0)leave->DIRECTION[i-1]
 | going[i]->DIRECTION[i] ).

BRIDGE_CONTROLER=(south.going[s:T] -> north.going[n:T] -> (
   when (n==0) south.enter -> BRIDGE_CONTROLER
 | when (s==0) north.enter -> BRIDGE_CONTROLER)
)+{south.enter,south.going[T], north.enter,north.going[T]}.

||BRIDGE=(north:DIRECTION || south:DIRECTION || BRIDGE_CONTROLER).
```

Design a concurrent implementation of this bridge using UML or OMT class diagrams for an implementation that uses Java threads. Give a rationale for the relationships you have chosen.

[9 marks]

c. Java method calls are synchronous. This means that a method m that calls another method n is blocked while n is being executed. With an asynchronous model, m would continue to execute as soon as n has accepted the call and m and n synchronise later. Asynchronous behaviour is often desirable in a distributed setting. Explain how Java threads can be used to implement asynchronous method calls. Discuss how the asynchronous call can be made and what options are available for synchronisation.

[8 marks]

[Total 33 marks]

4. The following shared-memory model for concurrent execution of programs has been suggested, where the shared memory is called the *pool*. The program behaviour is modelled as a game between *programmer* and *computer* as follows, where an *atom* (also refereed to as a *program statement*) is a simple instruction or a test condition, and a *packet* is a set of atoms. The programmer makes the first move and then play alternates between the computer and the programmer. The programmer in their turn submits a packet of atoms to the pool for eventual execution and the computer in their turn selects a single atom from the pool and then executes it. The programmer may at any stage submit the empty packet, denoted by *skip*, and may also submit *end* to denote that no more packets will be submitted. The computer may decide at any stage to *wait* instead of selecting an atom for execution.

   Answer the following three parts relating to above model.

   a. Show how sequential and concurrent execution of program statements can be enforced by the programmer.

   [11 marks]

   b. What would you consider to be *fair* play by the computer?

   [11 marks]

   c. Consider the following pseudo-code in a concurrent programming language, where **cobegin** (S_1) // . . . // (S_n) **coend** specifies the concurrent execution of a sequence $S_1, \ldots, S_n$ of programs:

      1. x := *true*; y := 1;
      2. **cobegin** (**while** x **do** y := y + 1) **end while** // (x := *false*) **coend**;

      Outline how the programmer should play to model the above program so as to ensure its eventual termination, given that the computer is playing *fairly*.

      [11 marks]

[Total 33 marks]

5. Answer the following three parts.

a. Define the notion of serialisability using an example of two transaction which are serialisable and another two transaction which are *not* serialisable.

[11 marks]

b. i. Can deadlock always be avoided in the the two-phase locking protocol ? Give an example to illustrate your answer.

[5 marks]

ii. Show how deadlock can be detected in the the two-phase locking protocol.

[6 marks]

[Total 11 marks]

c. In a client-server model, communication between a client and a server can be realised via the *remote procedure call* (RPC) model. In the synchronous RPC model the client executes a statement that calls a procedure, and then, prior to continuing execution, waits for the called procedure to be executed by the server and return control back to it.

Suggest some problems with this model, in the event of client or server failure, and how queues may be used to solve these problems.

[11 marks]

[Total 33 marks]

END OF PAPER