Answer the first question and two further questions.

1. a. Define the meaning of the following constructs of the Finite State Process (FSP) notation.

i. action prefix (”->”)

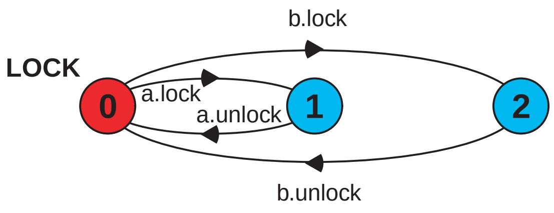
[3 marks]

ii. choice (”|”)

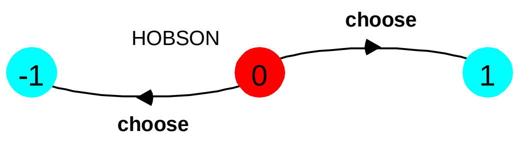
[3 marks]

[Subtotal 6 marks]

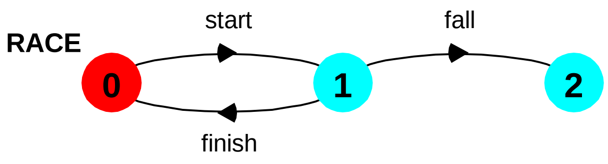
b. For each of the following Labelled Transition Systems (LTS), give an equivalent FSP specification.

i. 

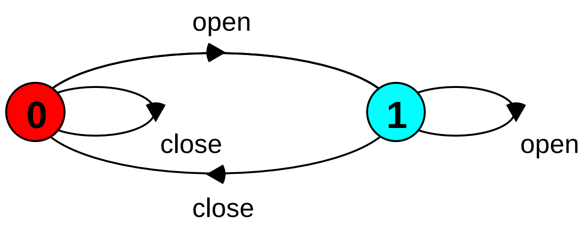
[4 marks]

ii 

[3 marks]

iii. 

[3 marks]

iv. 

[4 marks]

[Subtotal 14 marks]

TURN OVER

c. For each of the following FSP specifications, give an equivalent LTS.

i.
```
SQUARE = (in[i:1..2]->out[i*i]->SQUARE).
```
[3 marks]

ii.
```
CLOCK          = CLOCK[0],
CLOCK[i:0..4] = (when(i<4) tick->CLOCK[i+1]).
```
[4 marks]

iii.
```
DICE = (throw[i:1..6]->(when (i==6)again->DICE)).
```
[3 marks]

iv.
```
ALICE = (start->alice.finish->ALICE).
BOB   = (start->bob.finish->BOB).
||AB  = (ALICE||BOB).  //draw LTS for AB only
```
[4 marks]

[Subtotal 14 marks]

[Total 34 marks]

CONTINUED

2. a. Briefly explain how a guarded action in an FSP specification is translated into part of a Java program that implements that specification.

[10 marks]

b. In an automatic cable car system, each cable car has its own controller. The function of this controller is to ensure that a cable car only leaves the terminus when it is full of passengers. A cable car can hold a maximum of N passengers. After departure, the cable car arrives at the other end, all the passengers leave the cable car and new passengers may then board for another trip. The alphabet of the cable car is depicted below, together with a definition of the meaning of each action.

`board` a passenger boards the cable car.

`cardepart` the cable car departs. This action is delayed until the cable car is full.

`cararrive` the cable car arrives at the other end. This action is delayed until after departure.

Specify the behaviour of CABLECAR in FSP.

[12 marks]

c. Implement the CABLECAR specification from part b with the above three actions as monitor methods programmed in Java.

[11 marks]

[Total 33 marks]

3. a. Explain briefly how a resource, shared by a set of processes, can be modelled in FSP.

[8 marks]

b. The cheese counter in a supermarket is continuously mobbed by hungry customers. To restore order, the management installs a ticket machine which issues tickets to customers. Tickets are numbered in the range 1..MT. When ticket MT has been issued, the next ticket to be issued will be ticket numbered 1, i.e. the management install a new ticket roll. The cheese counter has a display which indicates the ticket number of the customer currently being served. The customer with the ticket with the same number as the counter display then goes to the counter and is served. When the service is finished, the number is incremented (modulo MT). Given the structure diagram depicted below for the cheese counter system, specify the behaviour of each of the processes (CUSTOMER, TICKET, COUNTER) and the composite process CHEESE_COUNTER in FSP.

[12 marks]

c. Implement the specifications for COUNTER and TICKET in Java.

[13 marks]

[Total 33 marks]

CONTINUED

4. a. i. Explain the terms safety property and liveness property with respect to concurrent programs.

[4 marks]

ii. Draw the Labelled Transition System for the following safety property:

```
property POLITE = (knock->enter->POLITE).
```

[4 marks]

[Subtotal 8 marks]

b. A lift has a maximum capacity of ten people. In the model of the lift control system, passengers entering a lift are signalled by an enter action and passengers leaving the lift are signalled by an exit action. Specify a safety property in FSP which when composed with the lift will check that the system never allows the lift that it controls to have more than ten occupants.

[8 marks]

c. Explain what is meant by the term deadlock in the context of concurrent programs and explain how LTS models can be used to check for deadlock.

[8 marks]

d. It is possible for the following system to deadlock. Explain why this deadlock occurs.

```
Alice = (call.bob -> wait.chris -> Alice).
Bob   = (call.chris -> wait.alice -> Bob).
Chris = (call.alice -> wait.bob -> Chris).

||S = (Alice || Bob || Chris) /{call/wait}.
```

The following model attempts to fix the problem by allowing Alice, Bob and Chris to timeout from a call attempt. Is a deadlock still possible? If so describe how the deadlock can occur and give an execution trace leading to the deadlock.

```
Alice = (call.bob -> wait.chris -> Alice
        | timeout.alice -> wait.chris ->Alice).
Bob   = (call.chris -> wait.alice -> Bob
        | timeout.bob -> wait.alice ->Bob).
Chris = (call.alice -> wait.bob -> Chris
        | timeout.chris -> wait.bob ->Chris).

||S = (Alice || Bob || Chris) /{call/wait}.
```

[9 marks]

[Total 33 marks]

CONTINUED

5. a. Briefly outline the two different ways of creating a new thread in Java.

[10 marks]

b. A Special Savings Building Society Account is permitted to have a maximum balance of M hundred pounds. Savers may deposit one hundred pounds at a time into the account up to the maximum. They may withdraw money in multiple units of a hundred pounds so long as the account is not overdrawn. The alphabet of the process that models the savings account is depicted below, together with a definition of the meaning of each action.

`range T = 1..M`

`deposit`	deposit one hundred pounds. This action is blocked if the balance would exceed M.
`withdraw[T]`	withdraw an amount in the range T hundred pounds. This action is blocked if sufficient funds are not available.

Specify the behaviour of ACCOUNT in FSP.

[12 marks]

c. Implement the ACCOUNT specification from part b with the two actions as monitor methods programmed in Java.

[11 marks]

[Total 33 marks]

END OF PAPER