IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2008

BEng Honours Degree in Computing Part II
MEng Honours Degrees in Computing Part II
MSc in Computing for Industry
BEng Honours Degree in Information Systems Engineering Part III
MEng Honours Degree in Information Systems Engineering Part III
BSc Honours Degree in Mathematics and Computer Science Part II
MSci Honours Degree in Mathematics and Computer Science Part II
BSc Honours Degree in Mathematics and Computer Science Part III
MSc in Computing Science
for Internal Students of the Imperial College of Science, Technology and Medicine

This paper is also taken for the relevant examinations for the Associateship of the City and Guilds of London Institute
This paper is also taken for the relevant examinations for the Associateship of the Royal College of Science

PAPER C223=MC223=I3.27

CONCURRENCY

Tuesday 6 May 2008, 10:00
Duration: 120 minutes

Answer THREE questions

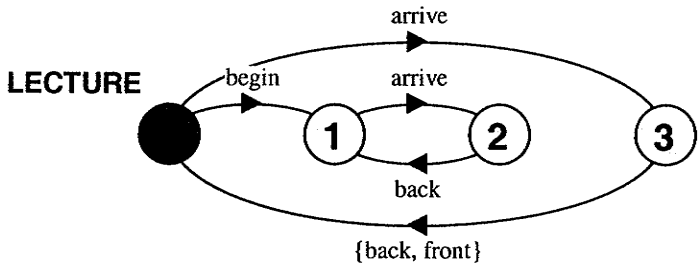
Paper contains 4 questions
Calculators not required

Section A *(Use a separate answer book for this Section)*

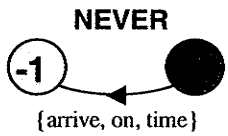
1a Explain how *non-deterministic choice* is expressed in Finite State Processes (FSP) modeling notation. Briefly explain why it is useful in modeling systems.

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

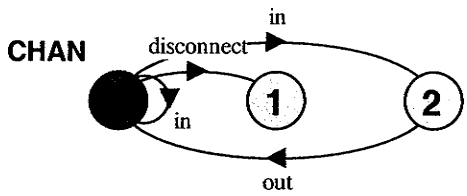
i)



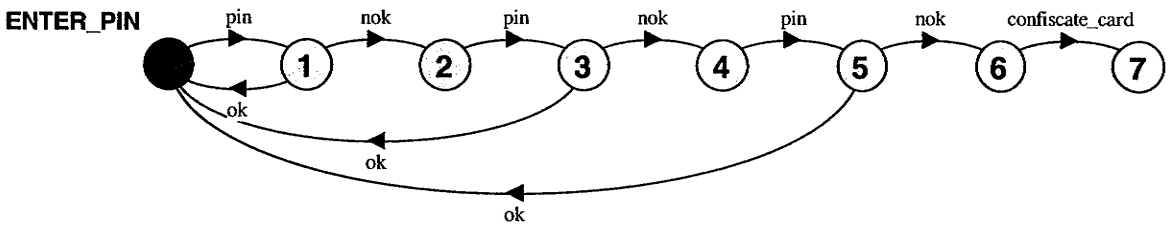
ii)



iii)



iv)



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

i)
```
CYCLE(N=4) = S[0],
S[i:0..3] = (out[i] -> S[(i+1)%N]).  // % is modulus
```

ii)
```
PERSON = (sleep -> dream -> PERSON
         |wake -> work -> PERSON).
INSOMNIA = STOP + {sleep}.
||SEATLE = (PERSON || INSOMNIA).  // draw LTS for SEATLE
```

iii)
```
EXIT     = (in -> out -> EXIT).
||OR = (a:EXIT || b:EXIT)/{in/{a.in,b.in}}.
// draw LTS for OR
```

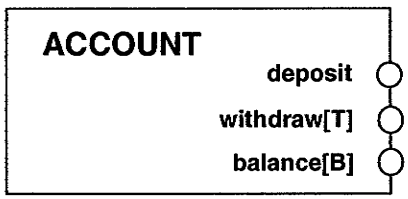
iv)
```
SQUARE = (in[i:1..3] -> out[i*i] -> SQUARE).
```

The three parts carry, respectively, 20%, 40% , 40% of the marks.

2a Briefly explain the concurrency problems that can arise when using nested monitors.

b A Special Euro Savings Society Account is permitted to have a maximum balance of **M** hundred Euros. Savers may deposit one hundred Euros at a time into the account up to the maximum. They may withdraw money in multiple units of a hundred Euros 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
range B= 0..M

deposit	- deposit one hundred Euros. This action is blocked if the balance would exceed **M**.
withdraw[T]	- withdraw an amount in the range **T** hundred Euros. This action is blocked if sufficient funds are not available.
balance[B]	- returns the balance in the range **B** hundred Euros.

Specify the behaviour of ACCOUNT in FSP.

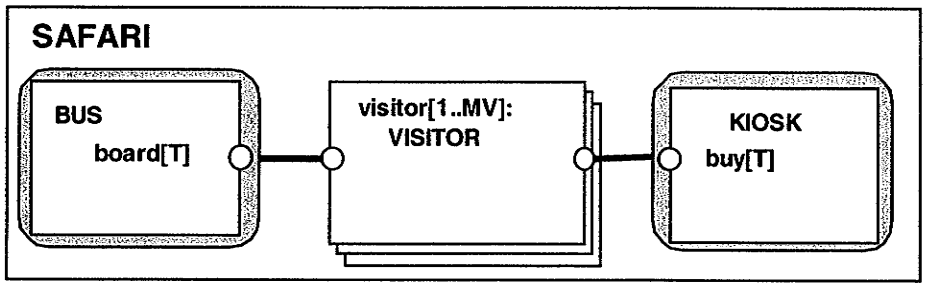
c Implement the ACCOUNT specification from part b with the three actions as monitor methods programmed in Java.

The four parts carry, respectively, 20%, 35%, and 45% of the marks.

Section B *(Use a separate answer book for this Section)*

3a Briefly outline the two different ways of creating a new thread in Java.

b Visitors to a Safari Park buy a numbered ticket from a kiosk. Tickets are numbered in the range **T = 1..MT**. When ticket **MT** has been issued, the next ticket to be issued will be ticket numbered 1, i.e. the ticket vendor installs a new ticket roll. After buying a ticket, the visitor proceeds to the bus stop to board the bus that tours the park. The visitor may only board a bus when his/her ticket number appears on a large display. The structure diagram for a model of this system with **MV** visitors is shown below:



Given that the behaviour of VISITOR is defined by:

```
VISITOR = (buy[t:T] -> board[t] -> STOP).
```

specify the behaviour of each of the processes (BUS, KIOSK) and the composite process SAFARI in FSP.

c Implement the specifications for each of the model entities (BUS, KIOSK, VISITOR) in Java. Include the definition of a method **`void build(int MV)`** which creates the objects required for SAFARI.

Briefly justify your use of `notify` or `notifyAll` where used.

The three parts carry, respectively, 20%, 30%, 50% of the marks.

4a Explain how it is possible for *safety property automata* in FSP to observe the correct execution of a model without constraining its behaviour.

Draw the Labelled Transition System for the following safety property:

```
property
TIMELY = (arrive -> TIMELY | begin_lecture -> LATE),
LATE = (end_lecture -> TIMELY | begin_lecture -> LATE).
```

and give *two* examples of traces that violate the property.

b Organisers of a party determined to ensure a good balance between girls and boys at the party have decided to implement a party entrance policy enforced by a matchmaker which only allows people into the party in pairs of one boy and one girl. Given the following definitions:

```
const N    = 4        //maximum size of boy and girl queue.
BOYS = (arriveBoy -> BOYS).
GIRLS = (arriveGirl -> GIRLS).
```

Specify a process **MATCHMAKER** in *FSP* that will **pair** one boy and one girl from queues of girls and boys forming at the party entrance while ensuring:
- the number of boys in the boys queue is always less than N, and
- the number of girls in the girls queue is always less than N.

c Given the process:

```
ROTATE  = (move -> LEFT | move -> RIGHT),
LEFT    = (up -> left -> LEFT | down -> left -> LEFT),
RIGHT   = (up -> right ->RIGHT| down -> right -> RIGHT).
```

State which of the following progress properties:

```
progress VERTICAL     = {up,down}
progress HORIZONTAL   = {left,right}
progress UP           = {up}
progress DOWN         = {down}
progress LEFT         = {left}
progress RIGHT        = {right}
```

are satisfied/violated in each of the following systems:

```
i)  ||ONE   = (ROTATE).
ii) ||TWO   = (ROTATE)>>{up}.
iii)||THREE = (ROTATE)<<{up}.
```

Give brief reasons for your answers with respect to the LTS for each system.

The four parts carry, respectively, 20%, 30%, 50% of the marks