CS3211 Parallel and Concurrent Programming – Guidelines for tutorial (22-26 March 2010)

Sample Exercises:
[Please conduct these as an interactive discussion, rather than an evaluation. Please also make it clear to the students that they are not being evaluated for their performance in these exercises, so that they are not afraid to make mistakes while answering.]

1. What action trace violates the following safety property?

   Property PS = (a -> (b->PS | a-> PS) | b->a->PS).

Answer:
```
/*
Trace to property violation in PS:
      b
      b
*/
```

2. A lift has a maximum capacity of ten people. In the model of the lift control system, passengers entering the lift are signalled by an enter action, and passengers leaving the lift are signalled by an exit action. Specify a safety property as a process equation which when composed with the lift will check that the system never allows the lift to have more than 10 occupants.

```
const N = 10

property LIFTCAPACITY = LIFT[0],
LIFT[i:0..10] = (enter -> LIFT[i+1]
                |when(i>0) exit  -> LIFT[i-1]
                |when(i==0)exit  -> LIFT[0]
                ).


Alternatively

Property LIFTCAPACITY = LIFT[0]
LIFT[i] = (when (i < 10) enter -> LIFT[i+1]
            | when (i >0) exit -> LIFT[i-1]
            | when (i==0) exit -> LIFT[0]
           ).

/*
Trace to property violation in LIFTCAPACITY:
      enter
      enter
      enter
      enter
      enter
      enter
      enter
      enter
      enter
      enter
      enter
*/
```

3. Recall the car park problem discussed in our lecture on monitors (Lec6.ppt). A controller is required for a carpark, which only permits cars to enter when the carpark is not full and does not permit cars to leave when there are no cars in the carpark. Car arrival and departure are simulated by separate threads.

```
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 =
        (ARRIVALS||CARPARKCONTROL(4)||DEPARTURES).
```

Specify a safety property which states that the car park does not overflow. Specify a progress property which asserts that cars eventually get to enter the car park. If car departure is lower priority than car arrival, does starvation occur?

Answer:

```
property OVERFLOW(N=4) = OVERFLOW[0],
OVERFLOW[i:0..N] = (arrive -> OVERFLOW[i+1]
                   |depart -> OVERFLOW[i-1]
                   ).

CHECK_CARPARK = (OVERFLOW(4) || CARPARK).

/* try safety check with OVERFLOW(3) */

progress ENTER = {arrive}

LIVE_CARPARK = CARPARK >>{depart}.
```

4. In an operating system, a binary semaphore is used to control access to the console. The console is used by user processes and system processes. Construct a model of this system and investigate the scheduling conditions under which user processes may be denied access to console.

```
BSEMA = (up -> down -> BSEMA).

PROCESS  = (console.up -> console.down -> PROCESS).

set Processes = {user[1..2],system[1..2]}

/* system processes have higher priority than user processes */
OS = ( Processes:PROCESS || Processes::console:BSEMA)>>{user}.
```