THE AUSTRALIAN NATIONAL UNIVERSITY

Mid Semester Examination, September 2012

COMP2310 / COMP6310
(Concurrent and Distributed Systems)

Writing Period: 1 hour duration

Study Period: 0 minutes duration

Permitted Materials: One A4 page with notes on both sides.
NO calculator permitted.

Questions are NOT equally weighted.
This exam will contribute 10% to your final assessment.

The questions are followed by labelled, framed blank panels into which your answers are to be written. Additional answer panels are provided (at the end of the paper) should you wish to use more space for an answer than is provided in the associated labelled panels. If you use an additional panel, be sure to indicate clearly the question and part to which it refers to.

The marking scheme will put a high value on clarity so, as a general guide, it is better to give fewer answers in a clear manner than to outline a greater number in a sketchy, half-answered fashion.

Please write clearly – if we cannot read your writing you may lose marks!

Student Number (please write clearly):							

Enrollment (circle one):
COMP2310
COMP6310

Official use only:

Q1 (12)	Q2 (14)	Q3 (20)	Q4 (4)	Total (50)

QUESTION 1 [12 marks]

(a) Define precisely under what circumstances two events (actions) should be considered to have occurred concurrently.

QUESTION 1(a) **[2 marks]**

(b) Briefly describe two motivations for using concurrent programming.

QUESTION 1(b) **[2 marks]**

(c) Describe in general terms what it means for a concurrent program to be correct. How does this differ from the notion of correctness for sequential programs?

QUESTION 1(c) **[4 marks]**

(d) Give two reasons why is can be useful to model a concurrent system before implementing it.

QUESTION 1(d) **[2 marks]**

(e) Can you always assume that concurrent programs will be executed concurrently on the hardware level? Give precise reasons.

QUESTION 1(e) **[2 marks]**

Additional answers to QUESTION 1 (__)[__]

QUESTION 2 [14 marks]

(a) A dice may repeatedly engage in a **throw** action followed by a **show** action, which has an integer value from 1 to 6 associated with it.

(i) Model such a dice using the FSP process `DICE`.

QUESTION 2(a)[i] **[2 marks]**

(ii) Specify an FSP process composition with two independent dice processes, || `DIE`.

QUESTION 2(a)[ii] **[2 marks]**

(iii) How many states has the processes `DICE`?

QUESTION 2(a)[iii] **[1 mark]**

(iv) How many states has the processes || `DIE`?

QUESTION 2(a)[iv] **[1 mark]**

Additional answers to QUESTION __(__)[__]

(b) Consider the following FSP process definition.

```
MAKE_A = (makeA -> ready -> used -> MAKE_A).
MAKE_B = (makeB -> ready -> used -> MAKE_B).
ASSEMBLE = (ready -> assemble -> used -> ASSEMBLE).
||FACTORY = (MAKE_A || MAKE_B || ASSEMBLE).
```

(i) Write the Labelled Transition System for **MAKE_A**.

QUESTION 2(b)[i] **[2 marks]**

(ii) Write *either* the Labelled Transition System for ||**FACTORY**, *or* an equivalent FSP process definition that does not use the || operator.

QUESTION 2(b)[ii] **[4 marks]**

(iii) Write all traces of length 5 for ||**FACTORY**.

QUESTION 2(b)[iii] **[2 marks]**

QUESTION 3 [20 marks]

(a) State two ways in which an **alive** Java thread may transition to the **terminated** state.

QUESTION 3(a) **[2 marks]**

(b) Briefly explain how two concurrent processes, neither one of which deadlocks on its own, can deadlock when run concurrently. Illustrate with a simple example.

QUESTION 3(b) **[4 marks]**

(c) The Java notification methods `wait()`, `notify()` and `notifyAll()` provide a form of condition synchronization. Semaphores also provide a form of condition synchronization. List two significant differences between these two approaches to condition synchronization.

QUESTION 3(c) **[2 marks]**

(d) Assume that there are three threads or processes, X, Y, and Z, that repeatedly and continuously print `X`, `Y`, and `Z` respectively. Use a synchronization primitive of your choice to coordinate the printing such that the number of `Z`'s printed is always less than or equal to the minimum of `X`s and `Y`s printed. Use pseudo-code or any programming language which you are familiar with.

QUESTION 3(d) **[8 marks]**

Question 3 (continued)

(e) Describe the term *lock* and explain how it might be implemented using an a atomic instruction.

QUESTION 3(e) **[4 marks]**

Additional answers to QUESTION ___(___)[___]

Student Number: .....

QUESTION 4 [4 marks]

(a) Describe two advantages the message passing paradigm has over the shared object paradigm

QUESTION 4(a) **[2 marks]**

(b) Give one advantage and one disadvantage of asynchronous message passing over synchronous message passing.

QUESTION 4(b) **[2 marks]**

Additional answers to QUESTION ___(___)[___]

Additional answers to QUESTION __(__)[__]

Additional answers to QUESTION __(__)[__]

Additional answers to QUESTION __(__)[__]