

THE AUSTRALIAN NATIONAL UNIVERSITY
Second Semester Examination – November 2009

COMP2310/6310/INFT4005

Concurrent and Distributed Systems

Study Period: 15 minutes

Time Allowed: 3 hours

Permitted Materials: non-programmable calculator

Questions are NOT equally weighted.

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.

More marks are likely be awarded for answers that are short and concrete than for answers of a sketchy or rambling nature. Answers which are not sufficiently legible may not be marked.

This exam is marked out of 100. You should answer all questions.

Your student number must be written at the top of every sheet of the exam paper. Write all answers using blue or black pen.

Student Number:

The following are for use by your friendly examiner.

Q1 Mark

Q2 Mark

Q3 Mark

Q4 Mark

Q5 Mark

Total Mark

Question 1 [40 marks] Content through to mid-semester

- (a) There are two processes, P and Q, that issue the following instructions on a single shared integer n:

```

integer n ← 1
Process P      Process Q
p1: n ← n + 1  q1: n ← n - 1
    
```

Discuss the possible final value of variable n, if the code is executed on a register based computer with a single processor.

[4 marks]

- (b) Detail two fundamentally different ways by which the Ada `select` statement can be used to express non-deterministic program behaviour.

[2 marks]

UID:

- (c) Give **five** basic characteristics of a monitor and state whether the Java and Ada programming languages provide monitors as a first order language construct.

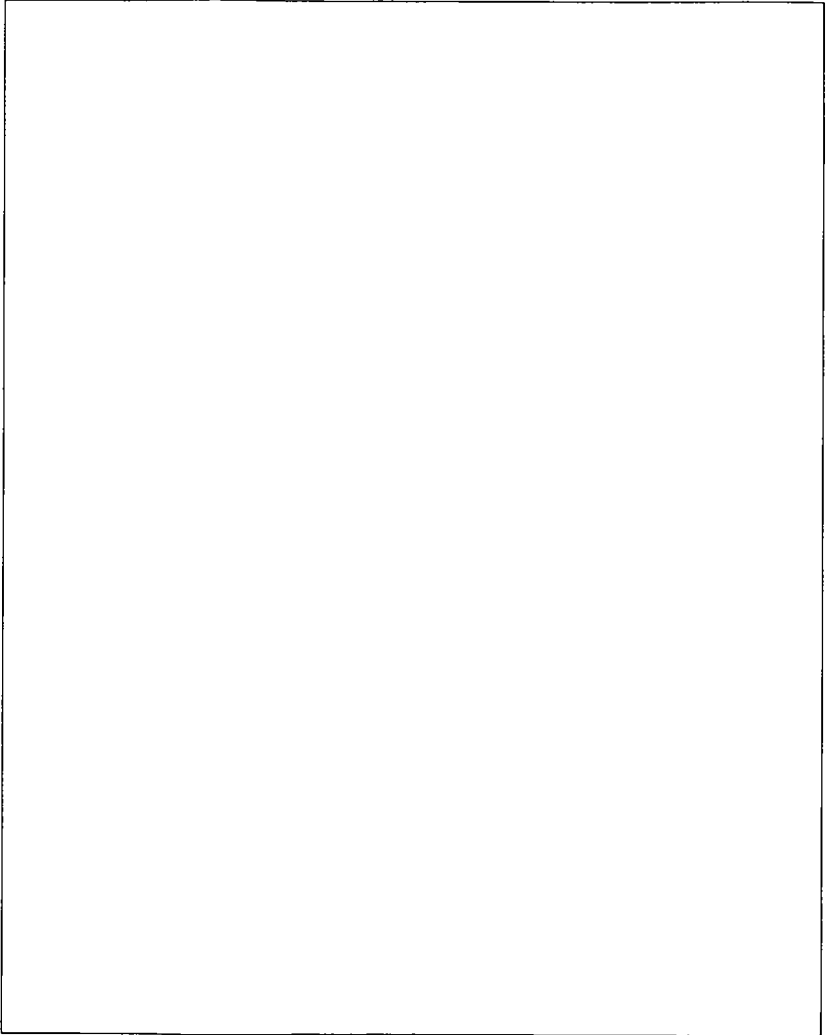
[3 marks]

- (d) Standard concepts of program correctness include partial and total correctness. Explain in words what is meant by each of these concepts AND why they may not be applicable to concurrent systems.

[3 marks]

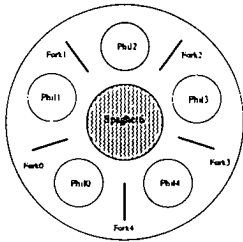
UID:

(e) Show how you would use a binary semaphore to construct a general semaphore.



[5 marks]

- (f) There are five philosophers sitting at a table with a bowl of spaghetti in the middle. There is a fork between each philosopher. A philosopher can only eat spaghetti if he/she has acquired both of the forks that are next to them. The philosophers engage in only two activities, thinking and eating. You write a program to model this scenario. Each fork is modelled as a semaphore array and each philosopher is initialized with its index i . Addition is implicitly modulo 5. A diagram of the philosophers, forks and spaghetti, with a copy of your initial code is given below:



```
semaphore array[0..4] fork ← [1,1,1,1,1]
```

```

loop forever
p1:  think
p2:  wait(fork[i])
p3:  wait(fork[i+1])
p4:  eat
p5:  signal(fork[i])
p6:  signal(fork[i+1])

```

The correctness properties are a) a philosopher eats only if he/she has two forks b) no two philosophers can hold the same fork at the same time c) freedom from deadlock d) freedom from starvation e) efficient behavior in the absence of contention.

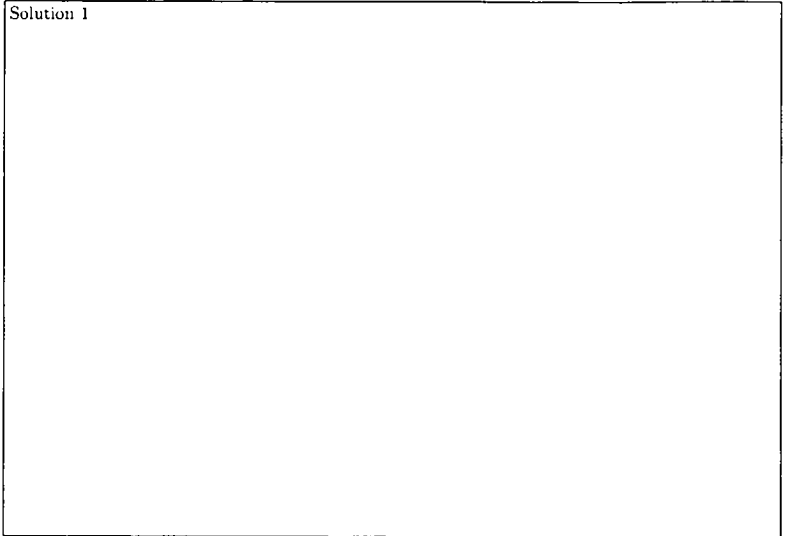
- (i) The code does not work correctly. For each correctness property state whether your program meets that criteria or not, and why, or what further information you require in order to address this issue.

[5 marks]

UID:

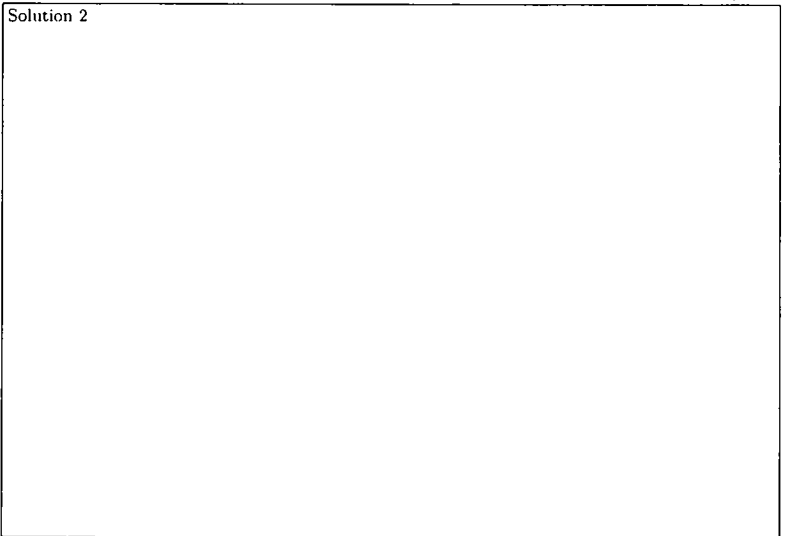
- (ii) Give TWO fundamentally different modifications to the code such that the modified code works correctly. Your modified codes must maintain the same number of philosophers and forks.

Solution 1



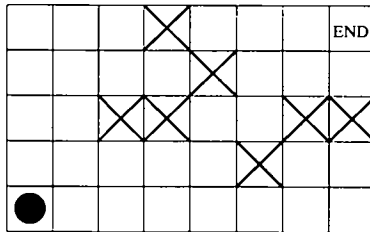
[4 marks]

Solution 2



[4 marks]

- (g) The diagram below shows a counter that has been placed in the lower lefthand square of a rectangular board. Some of the squares on the board are blocked (indicated by a cross). The objective is to determine if it is possible for the counter to move to the upper righthand square of the board by a sequence of moves either upward or to the right. (The counter cannot move downwards, to the left or diagonally. For the grid shown there exists a solution.)



You develop a concurrent algorithm that solves the above problem and plan to implement it using Ada. Outline the basis of your algorithm, the key Ada functionality you will use (eg select, request etc) and why. You are not required to write source code. You are expected to address issues such as when tasks are created and when they are destroyed, and to ensure that the program terminates in a suitable manner either because the objective has been met OR because it is impossible.

UID:

previous question continued

[10 marks]

Question 2 [16 marks]

Concurrent Code Analysis

- (a) The following Ada program is syntactically correct and will compile without errors or warnings:

```

with Ada.Text_IO; use Ada.Text_IO;

procedure Synchronize_It is
  task Stack_1 is
  begin
    loop
      select
        accept Push;
      or
        accept Pop;
      or
        terminate;
      end select;
    end loop;
  end Stack_1;

  protected Stack_3 is
    entry Push;
    entry Pop;
  private
    Filled : Boolean := False;
  end Stack_3;

  task Pop_Push;
  task Push_Pop is
    entry Stack2;
  end Push_Pop;

  protected body Stack_3 is
    entry Push when not Filled is begin
      Filled := True;
    end Push;
    entry Pop when Filled is begin
      Filled := False;
    end Pop;
  end Stack_3;

task body Pop_Push is
begin
  Stack_1.Pop; Stack_1.Push; Put_Line("Pop_Push done Stack_1");
  Push_Pop.Stack2;
  Stack_3.Pop; Stack_3.Push; Put_Line("Pop_Push done Stack_3");
end Pop_Push;

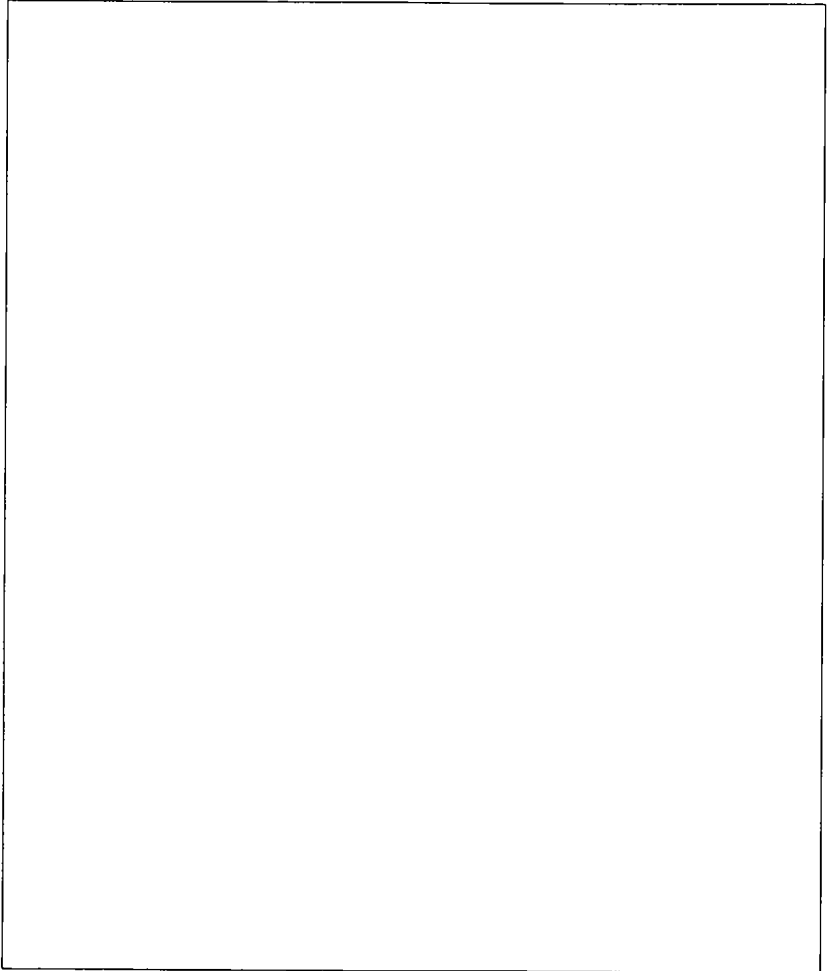
task body Push_Pop is
begin
  Stack_1.Push; Stack_1.Pop; Put_Line("Push_Pop done Stack_1");
  accept Stack2;
  Stack_3.Push; Stack_3.Pop; Put_Line("Push_Pop done Stack_3");
end Push_Pop;

begin
  null;
end Synchronize_It;

```

UID:

Consider the tasks `Pop_Push` and `Push_Pop`: which of them will terminate? If one or both of them will not terminate, how far do you expect them to get (i.e. what output do you expect to appear)? If you think you need to distinguish multiple cases then describe each case precisely. Does the whole program terminate? Is your answer dependent on the underlying machine architecture (single-processor, multi-processor) or operating system?



[10 marks]

(b) Consider the following C code carefully

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/wait.h>
int main(void){
    int array1[2];
    int *array2;
    int status;

    array2 = (int*) malloc(2*sizeof(int));

    array1[0]=100;
    array1[1]=200;
    array2[0]=300;
    array2[1]=400;

    printf("Initial addresses of arrays: %p %p\n",array1, array2);

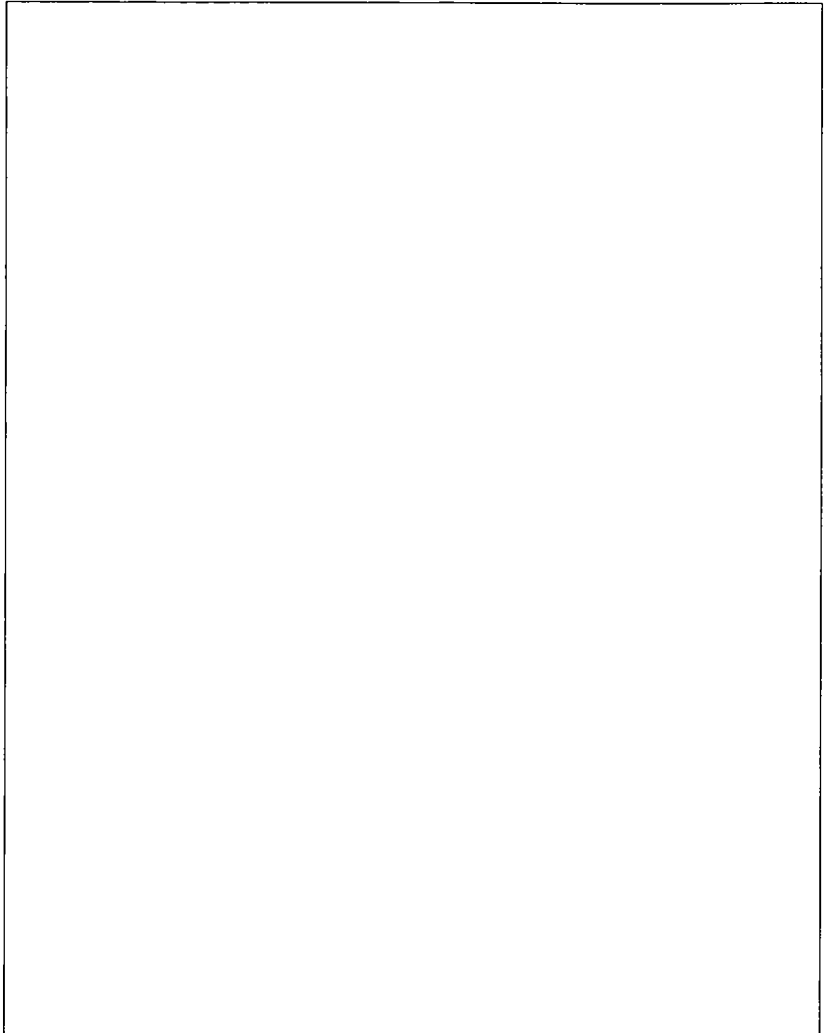
    if (fork() == 0){
        printf("FIRST array1 addr: %p holds %d %d\n",array1, array1[0],array1[1]);
        printf("FIRST array2 addr: %p holds %d %d\n",array2, array2[0],array2[1]);
        array1[1]=222;
        array2[1]=444;
    } else {
        printf("SECOND array1 addr: %p holds %d %d\n",array1, array1[0],array1[1]);
        printf("SECOND array2 addr: %p holds %d %d\n",array2, array2[0],array2[1]);
        wait(&status);
    }
    printf("THIRD array1 addr: %p holds %d %d\n",array1, array1[0],array1[1]);
    printf("THIRD array2 addr: %p holds %d %d\n",array2, array2[0],array2[1]);
    return 0;
}
```

The code compiles and runs without problems. The initial part of the output on the terminal reads (%p is a pointer type printing out the address in hex-code):

Initial addresses of arrays: 0xbfbf51b4 0x804a008

UID:

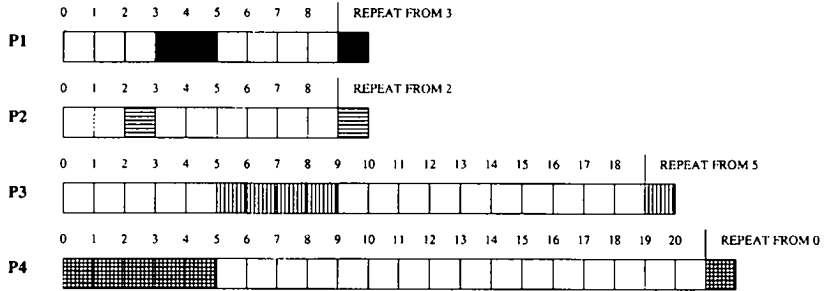
Detail the remaining output produced by the rest of the code. If you are unable to specify the output exactly, state why. Explain what you assume in your answer about the underlying runtime system.



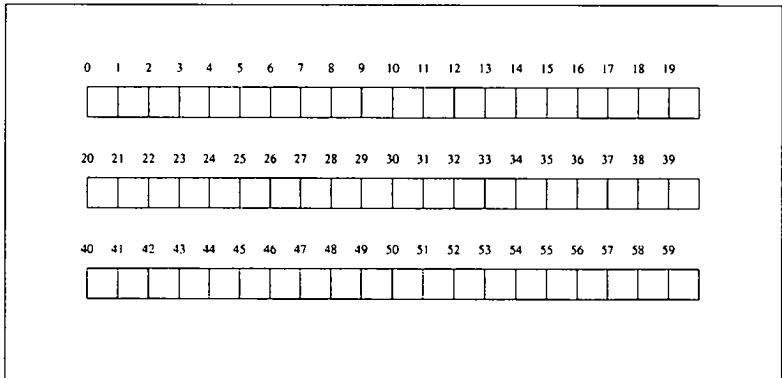
[6 marks]

Question 3 [10 marks] Scheduling

There are four processes (P1-P4) executing on a system with a single CPU. The CPU requirement of each process repeats with a certain period. These requirements are illustrated below, where shaded boxes indicate a CPU requirement, and blank boxes indicates no CPU requirement.



- (a) Complete the following timeline showing how the various CPU requirements of the different processes would be scheduled using first-come first-served scheduling. Do this by writing process numbers in boxes. Continue numbering boxes until EITHER the scheduling pattern repeats, OR until you run out of boxes. If the pattern repeats clearly mark on the diagram where it repeats.



[4 marks]

UID:

- (b) Repeat the above for the first 20 time slices, using feedback scheduling with 2^2 pre-emption intervals and a timeslice of 2 unit.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	

[3 marks]

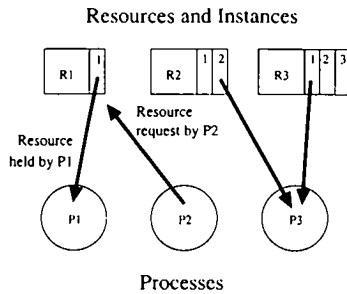
- (c) In the absence of process 3, would the CPU requirements of the remaining 3 processes be schedulable using rate monotonic fixed priority scheduling? Explain fully how you derive your answer.

[3 marks]

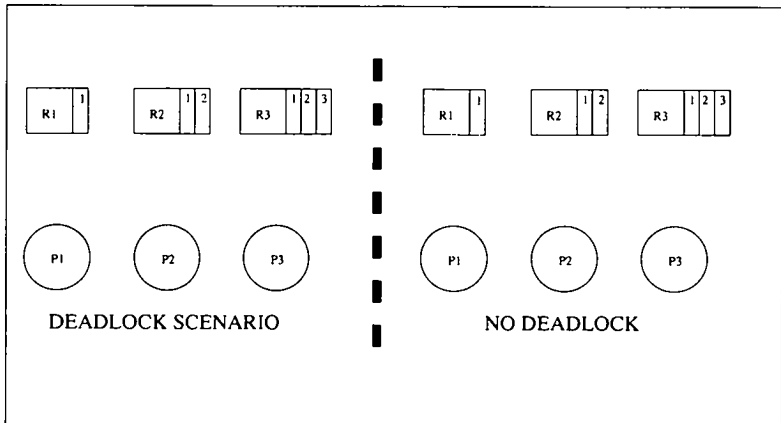
Question 4 [12 marks]

Safety and Liveness

- (a) In the following diagram there are three resources, R1, R2 and R3, and three processes, P1, P2 and P3. There is one instance of R1, two instances of R2 and three instances of R3. Each instance of a resource must be used under mutual exclusion. The arrows indicate that one instance of R1 has been allocated to process P1, that P2 is waiting on R1, and that one instance of R2 and R3 has been allocated to process P3.



At a certain point in their operation each process requires one instance of each resource type in order to progress. Allocating all resources, and using the same notation as above, annotate the two diagrams below to illustrate one deadlock scenario and one non-deadlock scenario.



[4 marks]

UID:

- (b) Consider the case where there are 4 resource types R1-R4 and five processes P1-P5. The total available resources of each type, the allocation of resources to processes, and the total resource requirements of the five processes are given below.

Total instances for each resource			
R1	R2	R3	R4
6	7	12	12

Proc	Allocated				Total Required			
	R1	R2	R3	R4	R1	R2	R3	R4
P1	2	0	0	0	2	7	5	0
P2	0	0	3	4	6	6	5	6
P3	1	0	1	2	2	1	1	3
P4	1	2	4	4	4	3	5	6
P5	0	3	3	2	0	6	5	2

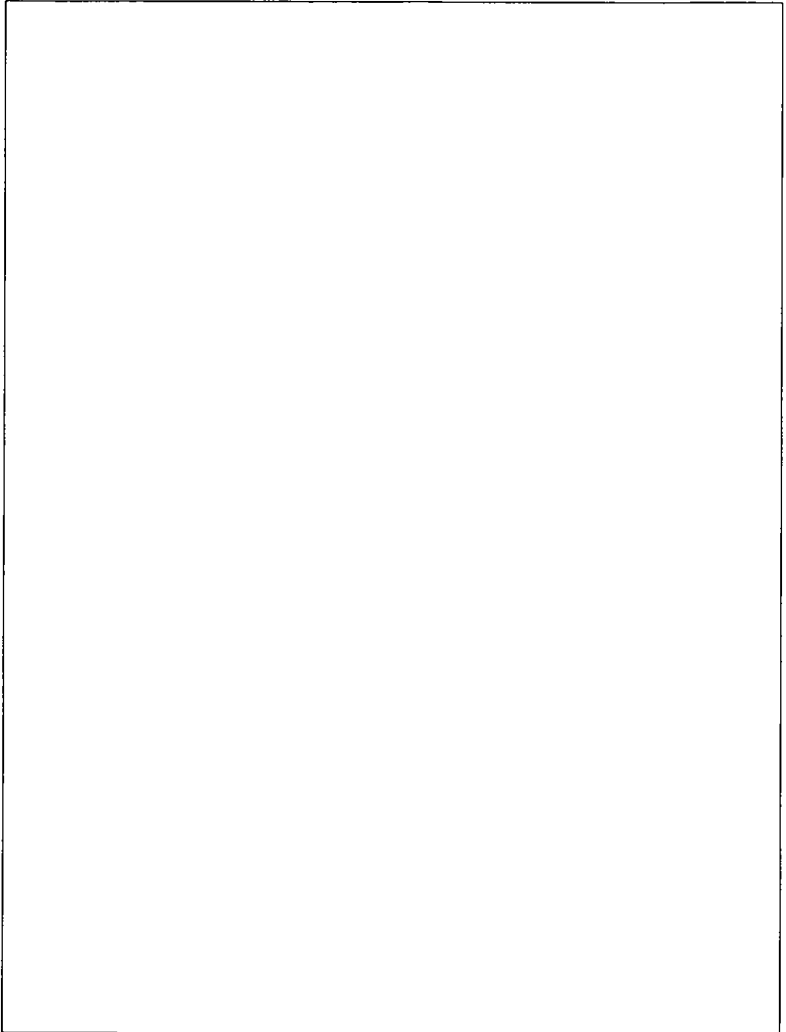
- (i) Use Bankers algorithm to analyse the above and determine whether the system is already in deadlock, or is in a state that will inevitably lead to deadlock.

[4 marks]

UID:

- (ii) If the system is currently deadlocked or going to deadlock, determine whether it is possible to remove the deadlock by having just one process relinquish control of its currently allocated resources.

If the system does not currently deadlock, trace out a series of future allocations that will lead to deadlock.



[4 marks]

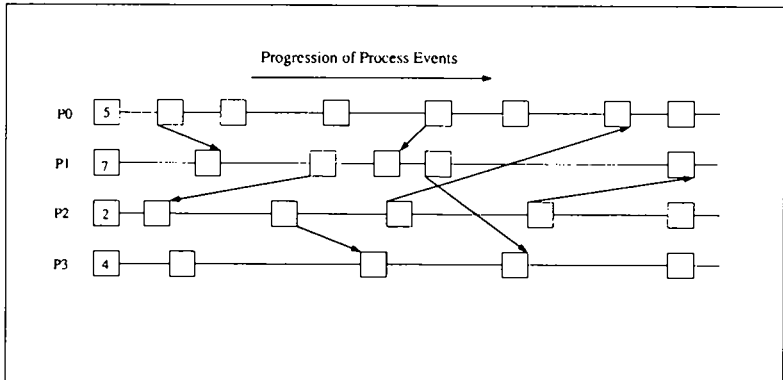
Question 5 [22 marks]

Distributed Systems

- (a) Why is it hard to implement synchronized clocks on a distributed system.

[2 marks]

- (b) The following diagram shows a system of four processes which are using Lamport's algorithm for logical time. The value of the logical time on each system is shown in the first square box of each process. Timing events are indicated by subsequent square boxes. These events may or may not be associated with inter-process message passing. Within each square box add the value of the local logical clock.



[4 marks]

- (c) In the Byzantine Generals algorithm there are four generals with names Basil, John, Leo and Zoe. They need to agree whether to attack (A) or retreat (R). Exactly one general is a traitor. The following shows the data structure associated with Zoe. Values labelled X_1 are to be defined.

General	Reported by				Majority
	Basil	John	Leo	Zoe	
Basil	R	A	R	-	X_1
John	R	A	A	-	X_2
Leo	R	R	R	-	X_3
Zoe	-	-	-	A	A
Overall Majority					X_4

- (i) Who is the traitor? Show clearly how you derive your answer

[3 marks]

- (ii) What are the values for X_1 , X_2 , X_3 and X_4 ?

[2 marks]

- (iii) Construct a minimal scenario leading to the above data structure.

[5 marks]

(iv) For this scenario complete the data structures below for the other two loyal generals

Data structure for second loyal general

Loyal General Name:					
General	Reported by				Majority
	Basil	John	Leo	Zoe	
Basil	R				
John		A			
Leo			R		
Zoe				A	
Overall Majority					

Data structure for third loyal general

Loyal General Name:					
General	Reported by				Majority
	Basil	John	Leo	Zoe	
Basil	R				
John		A			
Leo			R		
Zoe				A	
Overall Majority					

[6 marks]

UID:

Continuation of answer to Question Part

Continuation of answer to Question Part

UID:

Continuation of answer to Question Part

Continuation of answer to Question Part