Seminar 4 problem Sheet

1. Consider the following process set:

Process	**Priority**	**Release Time**	**Execution Pattern**
A	3	2	ERRE
B	2	5	EEEEE
C	1	0	ERRRRRE

(*An alternative way of representing the above information is as follows*

A = {1P(R)2V(R)1} released at time 2
B = {5} released at time 5
C = { 1P(R)5V(R)1} released at time 0

where P(R) represents the lock operation on resource R and V(R) the unlock operation. Each number in the brackets is the execution time for each piece of the task.}

where E represents the process executing without the resource and R the process executing with the resource.

a. Draw the time line for the process set.
b. Re-draw the time line when priority inheritance is used.

c. Calculate the blocking time for each process.

The Completion Time Theorem still holds but with the following modification, we now calculate the response time using

$$R_i = C_i + B_i + I_i$$

where C_i is the worst case cpu requirement, B_i is the time lost due to blocking, and I_i is the interference due to higher priority processes.

d. Test the schedulability of the task set.

An alternative approach to controlling priority inversion is to implement a policy that a process may not be pre-empted while in its critical section.

e. Draw a time line showing how the above process set would execute under this policy.

2. Consider the following process set:

Process	**Priority**	**CPU Time**	**Release Time**	**Resource Q (after, for)**
A	3	5	6	2,2
B	2	7	2	2,4
C	1	6	0	1,4

   a. Draw a time line for the above process set using simple priority pre-emption.
   b. Assume the deadlines for each process are 14,17 and 18 respectively do the processes make their deadlines?
   c. Suppose process C only requires resource Q for 2.5 units of time. Redraw the time line.
   d. Do the processes still make their deadlines?
   e. Re-work each scenario using
      i. basic priority inheritance;
      ii. Using the Ceiling Priority Protocol;
      iii. Using the Immediate Ceiling Priority Protocol;

3. Consider the following process set:

Process	**Priority**	**CPU Time**	**Release Time**	**Resource Q After,For**	**Resource R After,For**
A	4	5	4	1,2	3,1
B	3	6	2	3,2	1,2
C	2	3	2	1,2	
D	1	6	0	1,4	

(
A = {1P(Q)2V(Q)P(R)1V(R)1} released at time 4; priority 4 (high)
B = {1P(R)2V(R)P(Q)2V(Q)1} released at time 2; priority 3
C = {1P(Q)2VQ)} released at time 2; priority 2
D = {1P(Q)4V(Q)1} released at time 0; priority 1 (low)
)

Draw time-lines for the above for
   a. A simple priority system;
   b. Using basic priority inheritance;
   c. Using the Ceiling Priority Protocol;
   d. Using the Immediate Ceiling Priority Protocol;

Calculate the blocking times for each process under basic priority inheritance and under CPP.

4. Consider the following process set:

   A = {1P(Q)1V(Q)1} release time 7; priority 5 (high)
   B = {1P(R)1V(R)1} release time 5; priority 4
   C = {2} release time 4 ; priority 3
   D = {1P(Q)2P(R)1.5V(R)0.5V(Q)1} release time 2 ; priority 2
   E = {1P(R)4V(R)1} release time 0; priority 1 (low)

   Draw a time line to show the behaviour of this set when the scheduler uses priority inheritance.
   Assuming that the scheduler implements the Ceiling Priority Protocol what are the ceiling priorities of Q and of R? Draw a time line showing the behaviour of the system under the Ceiling Priority Protocol. Draw a graph showing the values of the priority ceiling for the system during the time the process set runs.
   Re-work the problem using the Immediate Ceiling Priority Protocol.

   Calculate the blocking times for each process under basic priority inheritance and under CPP.