CIS 520 – Quiz #1 – Fall 2011 Name: ___________________

Date: Wed., Oct. 12, 2011
Total: 100 pts

1. (25 points) Including the initial parent process, how many processes are created when the following program is executed? ________. Draw the Process Model showing the parent-child hierarchy, also indicate what will be printed by each process.

```
#include <stdio.h>
#include <unistd.h>

int main()
{
    int x = 1;

    /* fork off a child process */
    if (fork()==0)
     x++;

    /* fork off another child process */
    if (fork()==0)
     x++;

    printf("x = %d : ", x); fflush(stdout);

    /* wait for a signal to be received */
    pause();
}
```

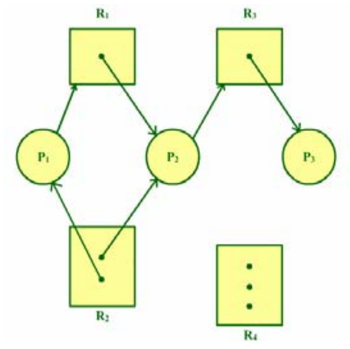
Which of the following outputs are possible? Circle all that apply:

a. `x = 1 : x = 2 : x = 3 : x = 3 :`
b. `x = 1 : x = 2 : x = 2 : x = 3 :`
c. `x = 1 : x = 2 : x = 2 :`
d. `x = 3 : x = 2 : x = 1 : x = 2 :`
e. `x = 3 : x = 2 : x = 1 :`
f. `x = 2 : x = 2 : x = 2 :`
g. `x = 1 :`
h. `x = 2 : x = 1 : x = 2 : x = 2 :`

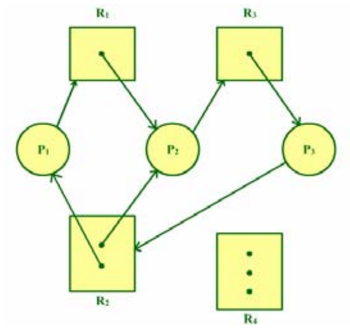
2. (25 points) Deadlock questions:

   a. If you have a system in which there is **no cycle** in the wait-for graph of the system. Is this a definitive sign of **no** deadlock? _______ Explain briefly.

b. Consider the following Resource-Allocation Graph. Is the system, with three process P_1, P_2, and P_3, deadlocked? ________ Explain briefly.



c. Suppose that a request for resource R_2 is made by P_3 as shown below. Is the system deadlocked? ________ Explain briefly.



d. Upon execution, can the following program deadlock? _______.
Assume that all system calls are successful.

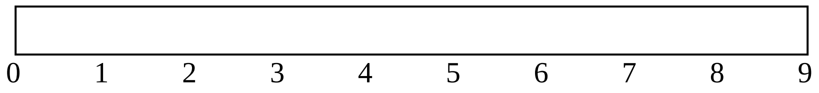
```
sem_t S;

void user_thread_code(void *_) {
  while(1) {
    // get two units of S
    sem_wait(&S);      // sema_down, P, wait ..
    sem_wait(&S);      // sema_down, P, wait ..
      // use resource in critical section .. code omitted
    sem_signal(&S);    // sema_up, V, signal ..
    sem_signal(&S);    // sema_up, V, signal ..
  }
}
int main(void)
{
  pthread_t t1, t2, t3;
  sem_init(&S, 3);  // S represents a resource with 3 units
  pthread_create(&t1, NULL, user_thread_code, NULL);
  pthread_create(&t2, NULL, user_thread_code, NULL);
  pthread_create(&t3, NULL, user_thread_code, NULL);
  ..
}
```

e. If so, give a specific execution sequence that leads to deadlock. If not, explain briefly why deadlock is not possible.

3. (25 points) Suppose that the following processes arrive for execution at the times indicated. Each process will run for the amount of time listed. In answering the first three parts, (a. – c.), use **non-preemptive scheduling** and base all decisions on the information you have at the time the decision must be made, for the last two parts, (d. – e.), use **preemptive scheduling**.

Process	Arrival Time	Run Time
P1	0.0	5.0
P2	1.0	3.0
P3	3.0	1.0

a. What is the average turnaround time for this set of processes if the First-Come, First-Served (FCFS) Scheduling Algorithm is used and processes are scheduled non-preemptively? Complete the Gannt Chart to show how the processes would be scheduled.



b. What is the average turnaround time for this set of processes if the Shortest Job First (SJF) Scheduling Algorithm is used and processes are scheduled non-preemptively?

c. Is the schedule generated using SJF optimal for non-preemptive processes? _____. If it is not optimal, generate another non-preemptive schedule that results in a smaller average turnaround time; e.g., draw a Gannt Chart showing an optimal schedule. Otherwise, explain briefly why SJF is always optimal.

d. What is the average turnaround time for this set of processes if the First-Come, First-Served (FCFS) Scheduling Algorithm is used and processes are scheduled **preemptively**?

e. What is the average turnaround time for this set of processes if the Shortest Job First (SJF) Scheduling Algorithm is used and processes are scheduled **preemptively**?

4. (25 points) Static priority scheduling: Assume you have a system with a priority-based CPU scheduler. Assume that the scheduler supports preemption but not priority donation. Assume that the priorities are set such that thread T2 has a high priority, thread T1 has the middle priority, and thread T0 has the low priority. Assume the system starts with only thread T0 executing T0( ). Assume that the semaphore **`S0`** is initialized to 1, and that waiting processes are enqueued in the wait queue of semaphores in FIFO order if priority donation is not used.

```
sem_t S0;
sem_init(&S0, 1); // initialize semaphore count to 1

void T0() {              void T1() {              void T2() {
 printf("T0-1\n");        printf("T1-1\n");        printf("T2-1\n");
 sem_wait(&S0);           StartThread(T2);         sem_wait(&S0);
 StartThread(T1);         printf("T1-2\n");        printf("T2-2\n");
 printf("T0-2\n");        sem_wait(&S0);           sem_signal(&S0);
 sem_signal(&S0);         printf("T1-3\n");        printf("T2-3\n");
 printf("T0-3\n");        sem_signal(&S0);        }
}                         printf("T1-4\n");
                         }
```

a. What is output when the code is executed? Clearly indicate if different interleavings (orderings) are possible.

b. What changes, if any, would occur in the output if priority donation was added to the scheduler?

c. Can the threads deadlock if priority donation is not used? _____ Can the three threads deadlock if priority donation is used? _____ Explain briefly.

d. Does priority donation prevent deadlock in all synchronization problems when processes are assigned different priorities? If not, give an example where priority donation would not help. If so, explain briefly why priority donation always helps.