CIS 721 - Real-Time Systems **Name: _____**
Final Exam
Due: Wednesday, Dec. 16, 6:00 pm (turn in paper and uploads by Thursday, Dec. 17, 12:00 noon)

1. (25 pts) Consider the task set consisting of the following three preemptive, periodic tasks (denoted using the notation $\tau_i = (\varphi_i, p_i, e_i, D_i)$) with φ_i = phase, p_i = period, e_i = run-time, and D_i = deadline of task τ_i.

   τ_1 = (0, 20, 4, 20)
   τ_2 = (10, 20, 6, 10)
   τ_3 = (10, 40, 9, 30)

   The system is to be scheduled and executed using a static cyclic schedule with a frame size of f = 10.

   (a) Draw a network flow graph that can be used to find a static cyclic schedule of the tasks. Remember to indicate maximum flows on all edges in the graph.

   (b) Compute a solution that maximizes flow and draw a Gannt Chart showing the corresponding schedule for jobs in the three tasks over one hyperperiod. What is the maximum total flow? _____

   (c) Could the same solution be used if the tasks were non-preemptive? _____ Explain briefly. If not, can the input file to the maximum network flow algorithm be modified to generate a feasible schedule for the same **non-preemptive** task set? _____ Describe the modifications required to the input file, or explain why such a modification is not possible.

   (d) Upload a copy of your original input, as ***original.input***, used to solve (b) and your modified input, as ***modified.input***, used to solve (c) above.

2. (25 pts) A system consists of three preemptive, periodic tasks, A, B, and C, with the following properties:

Task	Run Time (C_i)	Period (T_i)	Deadline (D_i)	Priority (low, medium, high)
A	2	6	6	
B	6	12	12	
C	2	18	9	

The tasks are scheduled using preemptive, priority-based scheduling, with different priorities assigned using a deadline-monotonic priority assignment.

(a) Specify the priority level from high to low in the table above.

(b) What is the total utilization (U) for these three tasks? _____ Show work.

(c) Can a Utilization-Based Test be used to ensure that the task set is schedulable? _____
If it can be applied, what conclusions, if any, can be made? Explain briefly.

(d) Does Response Time Analysis ensure that the task set is schedulable? _____ Show work.

(e) If all three tasks are non-preemptive, is the task set still schedulable using a priority-based scheduling algorithm with the same priority assignment as above? Explain briefly.

3. (25 pts) Consider the Promela model of an algorithm for enforcing mutually exclusive access to a critical section, given below. Use SPIN to answer the following questions:

   (a) Can **`turn[i]`** reach the value 255? How would you check?

   (b) Does the algorithm correctly enforce mutual exclusion? If not, briefly describe the sequence of events that leads to a violation of mutual exclusion.

```
byte turn[2]; /* who's turn is it? */
byte mutex;   /* number of processes in critical section */

active [2] proctype P() /* two processes compete */
{
    bit i = _pid;
    assert(i == 0 || i == 1);
Loop:
    turn[i] = 1;
    turn[i] = (turn[1-i] + 1)%256;
    /* wait for the following condition to become true */
    (turn[1-i] == 0) || (turn[i] < turn[1-i]);
CS: /* the critical section */
    mutex++;
    assert(mutex == 1); /* mutual exclusion check */
    mutex--;
    turn[i] = 0;
    goto Loop
}
```

   (c) Suppose that the statement "**`turn[i] = (turn[1-i] + 1)%256;`**" is replaced with the following if statement. Now, does the algorithm correctly enforce mutual exclusion? If not, briefly describe the sequence of events that leads to a violation of mutual exclusion, and modify the if statement so that mutual exclusion is ensured.

```
if
:: (turn[1-i] == 255) -> turn[i]=1; turn[1-i]=0;
:: (turn[1-i] < 255) -> turn[i] = turn[1-i] + 1;
fi;
```

4. (25 pts) Solve the problem called **Non-trivial Quibble** in Martin Gardner's book, "aha! Insight", pages 7-8, using UPPAAL. Begin with a row of n objects of one type, adjacent to n objects of another type. You wish to change the row to have alternating objects by making a set of legal moves. A move is legal if you slide any adjacent pair of counters to any open position on the row, without altering the order of the two objects that are moved.

   For example, suppose that the objects consist of 3 X's and 3 O's; that is, n = 3. The steps required to solve the puzzle are shown below:

```
XXXOOO
  XOOOXX
  XOO   XOX
    OXOXOX
```

   Use UPPAAL to try to find a solution when n is any value less than 6. It is trivial when n = 1. Hint: Allow moves to be made to both left and right of the original configuration.

```
XO
```

   You should find that the puzzle is not solvable when n = 2. For any n greater than 2, the puzzle can be solved in a minimum of n moves.

   (a) Specify the query used to prove this bound and the method used to find a solution using UPPAAL for the case n = 4.

   (b) Jot down the fewest steps required to solve the puzzle for n = 4 below:

```
XXXXOOOO
```

   (c) How many steps are required to solve the puzzle for n = 4 if the final configuration is XOXOXOXO?

   (d) Jot down the fewest steps required to solve the puzzle for n = 5 below:

```
XXXXXOOOOO
```

   (e) Upload a copy of your UPPAAL model as ***final.xml***. In all, you will have three files to upload, please put them in a single zip file, ***final2015.zip***. Thanks! Good luck!