Name: ______________________________ Student #: ________________

Engineering 8893
Concurrent Programming
Mid-Term Test

Dr. D. K. Peters

February 20, 2002

Instructions: Answer all questions. Write your answers on this paper. This is a closed book test, no textbooks, notes, calculators or other aides are permitted.
Total points: 50

---

1. [10 points] Consider the following pre-condition and statement:

   ```
   { x >= 2 } < x = x - 2; >
   ```

   For each of the following triples, show whether the above statement interferes with the triple:

   a) `{ x >= 0 } < x = x + 5; > { x >= 5 }`

   b) `{ x is odd } < x = x + 5; > { x is even }`

2. [10 points] Consider the following program:

   ```
   int x = 10;
   boolean c = true;
   co < await (x == 0) > ; c = false;
     // while (c) < x--; >
     // while (c) { if (x < 0) < x = 10; > }
   oc
   ```

   a) Is the program assured to terminate if the scheduling is weakly fair? Explain.

   b) Is the program assured to terminate if the scheduling is strongly fair? Explain.

3. [30 points] Suppose `N` processes share the use of `P` printers. Before using a printer, process `i` will call `request(i)`, which blocks until a printer is available. When a printer becomes available `request(i)` returns the identity (number) of the printer granted to that process. When the process is done with the printer, it calls `release(p)`, passing the printer number as the argument.

   a) [10 points] Give a the pseudo-code for a coarse-grained implementation (i.e., using conditional synchronization statements) for the `request` and `release` functions, including declarations and initial values for all shared variables. Your solution need not ensure fairness.

Name: ______________________ Student #: ______________

b) [15 points] Using the `Semaphore` class (as used in assignment #2, with methods `P` and `V`) for synchronization, give a pseudo-Java implementation of the the `request` and `release` functions and the class constructor (to give the initial values of all the variables).

c) [5 points] Outline how you would modify your solution to ensure that processes are granted access to printers in a first-come-fist-served order. How many semaphores would be required for your implementation?