Final Exam
12 December, 1990, 10:00 am-12:30 pm
14 December, 1990, 10:00 am-12:30 pm

Problem 1. (30 points) -- Linda meets Sort-Merge

You need to coordinate three processes. Two of the processes are *producers*, creating a sorted sequence of positive integers. At a very abstract level the producers execute the following program:

```
N := some-initial-value
do until N is big enough
     N := new-no-smaller(N)
     output N
when n is big enough
     output END-OF-SEQUENCE marker
```

The output of the two producers is read by a single *merger* process. This process will merge the two streams produced by the two producers until one sorted stream. For example, if one *producer* outputs the stream:

3 7 7 8 EOS

and the other *producer* outputs the stream:

3 6 8 EOS

the *merger* should output the stream:

3 3 6 7 7 8 8 EOS

Describe how you'd program these three processes using the Linda model of process creation and coordination.

Problem 2: (20 points) -- Vectorization

Show the data dependencies in the following code and describe how it might be efficiently vectorized.

```
     DO 100 I=2,200
     B(I) = B(I) + A(I)
     D(I) = D(I+1) + A(I+1)
100  A(I) = A(I-1) * B(I)
```

Problem 3. (20 points) -- Supercomputer management

Yesterday, you were hired as director of a supercomputer center. Last night you went to a bar, drank a bit too much, ran into a used computer salesman, and traded your center's Cray Y/MP for 213 used VAX 780's. After all a Y/MP is a 200 MIP machine and a VAX 780 is a 1 MIP machine....

Quick! Think of one problem that can be solved faster with 213 780's than with one Cray and write a convincing argument to give to your boss before he meets you for lunch today!

Problem 4. (30 points) -- OCCAM

Your task is to write an OCCAM process which receives values on two input channels:

PeekIn **PokeIn**

and produces values on one output channel:

PeekOut

Your process will serve as a shared memory of 10 registers.

The **PeekIn** and **PeekOut** channels are used in reading the shared memory. For example, if your process receives the value 5 on channel **PeekIn**, it should produce the present value of the 5'th register on channel **PeekOut**.

The **PokeIn** channel is used to set the shared memory registers. This is done in two "cycles". The first cycle gives a register address. The second gives a new data value. For example, if your process receives the value 4, followed by the value 1990, on channel **PokeIn**, the 4'th register is set to 1990.

Be sure your process reads its inputs and writes its outputs *as fast as possible*.