Name: ____________________________

Directions: MAKE SURE TO COPY YOUR ANSWERS TO A SEPARATE SHEET FOR SENDING ME AN ELECTRONIC COPY LATER.

1. (10) Fill in the blanks: We are considering using MPI or **snow** on a certain app, with 8 workers. For MPI, the number of sockets used at each worker will be [blank (a)] while for **snow** it will be [blank (b)].

2. Suppose we have partitioned matrices A and B, written in partitioned form:

$$\begin{pmatrix} A_1 & A_2 \\ 0 & A_3 \end{pmatrix} \tag{1}$$

$$\begin{pmatrix} B_1 & B_2 \\ 0 & B_3 \end{pmatrix} \tag{2}$$

Here each of the submatrices in A and B, including the one drawn as ",0" are $m \times m$.

(a) (20) Let $C = AB$, with

$$C = \begin{pmatrix} C_1 & C_2 \\ 0 & C_3 \end{pmatrix} \tag{3}$$

Express C_2 in terms of the submatrices in A and B. (In your electronic submission, use "A1" for A_1 etc. Use ' for transpose, and **use juxtaposition for multiplication**.

(b) (20) Assume all the A_i are invertible (you may not need all of them to be so), with $V_i = A_i^{-1}$. A^{-1} will then exist, and we'll write the inverse as

$$Q = \begin{pmatrix} Q_1 & Q_2 \\ 0 & Q_3 \end{pmatrix} \tag{4}$$

Show Q_2 in terms of the A_i and V_i. Again, write V_1 as "V1," etc.

3. (15) Consider the matrix-vector multiply **snow** code on p.22. If we were to convert it to a matrix-matrix multiply, everything in **mmul()** would pretty much carry over, except for the call to **Reduce()** in line 10. Show what the new contents of that line would be.

4. Consider the MPI code on pp.19ff. Suppose **N** is 10. (Note: The fact that this latter point is stated outside of (a) and (b) implies that it applies to both parts. Please keep this in mind in future quizzes.)

(a) (20) How many times will Node 1 call **MPI_Send()**?

(b) (15) Suppose in line 31 we were to accidentally write 1 instead of 0. What would happen? Choose one of the following:

(i) One of the nodes will have an execution error, e.g. divide by 0.

(ii) Two of the nodes will have an execution error.

(iii) Three of the nodes will have an execution error.

(iv) One of the nodes will hang.

(v) Two of the nodes will hang.

(vi) Three of the nodes will hang.

(vii) No node will have an execution error or hang, but the printed answer will be incorrect.

(viii) The correct answer will be printed out.

Solutions:

1.a 8 (7 worker connections, 1 manager)

1.b 1 manager connection

2.a $A_1 B_2 + A_2 B_3$

2.b We have

$$\begin{pmatrix} A_1 & A_2 \\ 0 & A_3 \end{pmatrix} \begin{pmatrix} Q_1 & Q_2 \\ 0 & Q_3 \end{pmatrix} = \begin{pmatrix} I & 0 \\ 0 & I \end{pmatrix} \tag{5}$$

Thus $A_3 Q_3 = I$, so $Q_3 = A_3^{-1}$.

We also have $A_1 Q_2 + A_2 Q_3 = 0$. Solving for Q_2, we get

$$Q_2 = A_1^{-1}(-A_2 Q_3) = -A_1^{-1} A_2 A_3^{-1} \tag{6}$$

3.

```
Reduce(rbind,mout)
```

4.a Node 0 will check values, 3, 5, 7 and 9 for divisibility by 3, with 5 and 7 surviving to go on to Node 1. The latter will take the 5 as **Divisor**, and then check 7 for divisibility by 5. The 7 survives this check and goes on to Node 2. That accounts for one call by Node 1 to **MPI_Send()**. Later it will make another such call, for END_MSG, thus a total of 2 calls.

4.b The PIPE_MSG messages become END_MSGs. Node 0 will send three of them to Node 1. The latter will use the first to set **Divisor**, and upon receiving the second, will send **Dummy** to Node 2. That node will use that message to set **StartDivisor**, then wait for a message to use for **ToCheck** — but that message will never come. So, the answer is (iv).