Name: ____________________________

Directions: **Work only on this sheet** (on both sides, if needed); do not turn in any supplementary sheets of paper. There is actually plenty of room for your answers, as long as you organize yourself BEFORE starting writing.

1. In the MPI code that finds prime numbers in a pipelined manner, let's say we measure work by the count of numbers checked by each node. (In the case of node 0, the even numbers won't count as being checked.)

(a) () Find the approximate ratio of the work done by node 0 and that of node 1, for large **N**.

(b) () Fill in the blank with a term from our course: The fact that the ratio in (a) is not near 1.0 shows that we have a problem with ______________________________.

2. () This problem also concerns the pipelined prime finding MPI code, in this case an altered version of it. Here we allow for a general number of nodes, rather than 3. Part of the code in **main()** changes to

```
   if (Me == 0) Node0();
   else if (Me == NNodes-1) NodeEnd();
        else NodeBetween();
```

Fill in the blanks for the code in **NodeBetween()**:

```
void NodeBetween()
{  int ToCheck,Dummy,Divisor;
   MPI_Status Status;
                                       // put 1 statement here
   while (1)  {
                                       // put 1 statement here
      if (Status.MPI_TAG == END_MSG) break;
      if (ToCheck % Divisor > 0)
                                       // put 1 statement here
   }
                                       // put 1 statement here
}
```

Solutions:

1. 3/2; load balancing

2.

```
void NodeBetween()
{  int ToCheck,Dummy,Divisor;
   MPI_Status Status;
   // first received item gives us our prime divisor
   // receive into Divisor 1 MPI integer from node Me-1, of any message
   // type, and put information about the message in Status
   MPI_Recv(&Divisor,1,MPI_INT,Me-1,MPI_ANY_TAG,MPI_COMM_WORLD,&Status);
   while (1)  {
      MPI_Recv(&ToCheck,1,MPI_INT,Me-1,MPI_ANY_TAG,MPI_COMM_WORLD,&Status);
      // if the message type was END_MSG, end loop
      if (Status.MPI_TAG == END_MSG) break;
      if (ToCheck % Divisor > 0)
         MPI_Send(&ToCheck,1,MPI_INT,Me+1,PIPE_MSG,MPI_COMM_WORLD);
   }
   MPI_Send(&Dummy,1,MPI_INT,Me+1,END_MSG,MPI_COMM_WORLD);
}
```