Please write your name on top of EVERY PAGE.

1. (36 pts) Short answer questions.

   (a) What is a process context switch?

   (b) Threads in the same process share memory, such as allocated memory on the heap and libraries. What state (in memory) is specific to each thread?

   (c) In Java, what guarantees does the declaration specifier volatile provide?

   (d) How does the Intel architecture present two hyperthreads on one physical core to the operating system?

   (e) What type of synchronization or concurrency problem was remotely debugged and patched from Earth to Mars on NASA's pathfinder mission?

2. (16 pts) OpenMP loop optimizations. Read both subparts first before answering question.

```
for (j=0; j<n; j++) {
  for (i=0; i<m; i++) {
    sum = sum + a[j*m+i]
  }
}
```

(a) Rewrite this code snippet to parallelize the loop using OpenMP and use an OpenMP reduction to eliminate the separable dependency among loops.

(b) Coalesce the loop to optimize your rewritten code. Hint: This conversion will allow OpenMP to realize balanced parallelism even when n is smaller that then number of cores in your machine.

3. (16 pts) On loop efficiency in the cache hierarchy

   (a) Why are the nested loops of the previous problem more efficient than the following loop that switches the iteration order.

```
for (i=0; i<m; i++) {
  for (j=0; j<n; J++) {
    sum = sum + a[j*m+i]
  }
}
```

   (b) Estimate the relative performance of the nested loop in the previous problem and the nested loop in this problem. Assume that a i and j and 64-bit integers and that L1 cache line contains 128 bytes. You should also assume tha the L3 cache is too small to hold n or m items. Show your work.

4. (16 pts) Initializing the MPI environment

   (a) What does each of the following function calls do?

   - MPI_Init (&argc, &argv)

   - MPI_Comm_size (MPI_COMM_WORLD, &num_procs)

   - MPI_Comm_rank (MPI_COMM_WORLD, &ID)

   (b) Why is it important for each process to know it's ID and number of processes? (This is a parallel pattern.)

5. 16 pts Draw a speedup chart for a parallel algorithm with an Amdahl number $p = 0.8$. Label the value of two specific points in your chart—typiically (1,1) and (x,y) that solves Amdahl's equation. What is the asymptotic speedup of this algorithm as available resources become infinite? Include dashed reference lines in your chart for ideal speedup and asymptotic performance.

Scratch space.