Programming Assignment 1: Due Friday, Sept. 14, 11PM MDT

To be done on water.eng.utah.edu (you all have accounts – passwords available if your CS account doesn't work)

1. Write a program to calculate π in OpenMP for a problem size and data set to be provided. Use a block data distribution.
2. Write the same computation in Pthreads.

Report your results in a separate README file.

- What is the parallel speedup of your code? To compute parallel speedup, you will need to time the execution of both the sequential and parallel code, and report speedup = Time(seq) / Time (parallel)
- If your code does not speed up, you will need to adjust the parallelism granularity, the amount of work each processor does between synchronization points. You can do this by either decreasing the number of threads or increasing the number of iterations.
- Report results for two different # of threads, holding iterations fixed, and two different # of iterations holding threads fixed. Also report lines of code for the two solutions.

Extra credit: Rewrite both codes using a cyclic distribution and measure performance for same configurations.

Programming Assignment 1, cont.

- A test harness is provided in pi-test-harness.c that provides a sequential pi calculation, validation, speedup timing and substantial instructions on what you need to do to complete the assignment.
- Here are the key points:
  - You'll need to write the parallel code, and the things needed to support that. Read the top of the file, and search for "TODO".
  - Compile w/ OpenMP: cc –o pi-openmp –O3 –xopenmp pi-openmp.c
  - Compile w/ Pthreads:

    cc –o pi-pthreads –O3 pi-pthreads.c –lpthread
  - Run OpenMP version: ./pi-openmp > openmp.out
  - Run Pthreads version: ./pi-pthreads > pthreads.out
- Note that editing on water is somewhat primitive – I'm using vim. You may want to edit on a different CADE machine.

Estimating π

$$\pi = 4\left[1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \cdots\right] = 4\sum_{k=0}^{\infty}\frac{(-1)^k}{2k+1}$$

```
double factor = 1.0;
double sum = 0.0;
for (k = 0; k < n; k++) {
    sum += factor/(2*k+1);
    factor = -factor;
}
pi_approx = 4.0*sum;
```