We noted in class today that since only one thread can enter a critical section at a time, the code in a critical section cannot be executed in parallel by the threads. At this point, spinlocks are our only mechanism for preventing multiple threads from entering a critical section at the same time.

For this assignment, you should look at how our simple spinlock affects performance. Modify the program `pth_pi_busy2a.c` so that it includes the header file `timer.h`. Use the `GET_TIME` macro at the points indicated by comments in the program, and add print statements to print the elapsed time of the Pthreads code and the serial code.

You should run your program with $n = 2^{20} = 1,048,576$, and `thread_count` = 2, 4, 8, 16, 32, and 64 on a single node of the penguin cluster. Report the run-times in your documentation. How does increasing the number of threads affect the overall run-time?

Note that the penguin cluster nodes only have four cores. So when you run the program with more than 4 threads, the operating system will need to switch between running threads, and all the threads can’t be run simultaneously.

Note also that there may be differences in the last two or three digits of the serial and parallel estimates. This is probably due to the use of floating point arithmetic and not to an error in the algorithm.