EECE5640
High Performance Computing
Homework 2
*Submit your work on Turnitin on Blackboard

1. (50) In 1965, Edsgar Dijkstra described the following problem. Five silent philosophers sit at a round table with bowls of noodles. Forks are placed between each pair of adjacent philosophers. Each philosopher must alternately think and eat. However, a philosopher can only eat noodles when she has both left and right forks. Each fork can be held by only one philosopher, and so, a philosopher can use the fork only if it is not being used by another philosopher. After she finishes eating, she needs to put down both forks so they become available to others. A philosopher can take the fork on her right or the one on her left as they become available, but cannot start eating before getting both of them. Eating is not limited by the remaining amounts of spaghetti or stomach space; an infinite supply and an infinite demand are assumed.

   Implement a solution for an unbounded odd number of philosophers, where each philosopher is implemented as a thread, and the forks are the synchronizations needed between them. Develop this threaded program in pthreads. The program takes as an input parameter the number of philosophers. The program needs to print out the state of the table (philosophers and forks) – the format is up to you.

   Answer the following questions: you are not required to implement a working solution to the 3 questions below, though adding each case to your C/C++ implementation will earn extra credit (on your quiz grade) for everyone who tries it.
   a.) Does the number of philosophers impact your solution in any way? How about if an even number of forks are used?
   b.) What happens to your solution if we give one philosopher priority over the rest?
   c.) What happens to your solution if we add a fork in the middle of the table?

   Also, discuss who was Edgar Dijkstra, and what is so important about this dining problem, as it relates to the real world. Also discuss the algorithm that bears his name, Dijkstra's Algorithm. Make sure to cite your sources carefully.
   *Answers to this question should be included in your homework write-up in pdf format, and submitted through Turnitin. You should include a C/C++ program submitted through Turnitin.

2. (30) In this problem, you will develop a program that computes prime numbers. You can refer to the Sieve of Eratosthenes for an example program. Evaluate the speedup that you achieve by using pthreads and multiple cores. You are free to use as many threads as you like. The program should take two input parameters, the number of threads, and the largest number. You will find all the primes between 1 and that number. Make sure to document the system you are running on and the number of hardware threads available.

*Answers to this question should be included in your homework write-up in pdf format, and submitted through Turnitin. You should include a C/C++ program submitted through Turnitin.

3. (20) In class we have learned about pthreads. These are also referred to as Posix threads. For the first part of this problem, discuss the differences between pthreads and processes. Second pthreads are considered "operating system-level" threads, versus green threads are considered "program-level" threads. Also, there are hardware threads. Describe the differences/similarities between these 3 forms of threading and provide an example of where they have been implemented in a hardware or software system. (This question is required for graduate students, but extra credit (on the quiz average) for undergraduates.)
   *Answers to this question should be included in your homework write-up in pdf format, and submitted through Turnitin.