Lab: Inter-Process Communication

Submit Assignment

Due Wednesday by 11:59pm **Points** 100 **Submitting** a file upload **File Types** zip
Available after Feb 19 at 10am

Lab Collaboration Policy

This is a team assignment. You will work with your teammate. No collaboration is allowed between different teams. Any sign of collaboration will result in a 0, and will be reported to the Honor Board.

Lab description

In our lecture we have explained the Dining Philosophers problem. Specifically, the Dining Philosophers problem is summarized as five philosophers sitting at a table doing one of two things: eating or thinking. While eating, they are not thinking, and while thinking, they are not eating. The five philosophers sit at a circular table with a large bowl of spaghetti in the center. A fork is placed in between each pair of adjacent philosophers, and as such, each philosopher has one fork to his left and one fork to his right. As spaghetti is difficult to serve and eat with a single fork, it is assumed that a philosopher must eat with two forks. Each philosopher can only use the forks on his immediate left and immediate right. In this lab, we will implement the Dining Philosophers problem via multi-thread programming by using mutex to control critical section.

Before Lab

Download skeleton code **labSkeleton.c**. The skeleton code provides a simple version of dining Philosophers problem where each philosopher takes the forks individually. The skeleton code has been explained in the lecture. The skeleton code uses the standard pthread package. The idea is to shield you from the irrelevant detail that inevitably is part of dealing with pthreads. This way, you use the standard package but you also focus on the lab at hand.

You can run “gcc -o labSkeleton labSkeleton.c **-lpthread**” to compile the skeleton code and get the executable file.

The skeleton code is implemented in C. We assume that you will do this lab in C. You are welcome to change the skeleton code to C++, as long as you can finish the lab.

In Lab

You will revise the skeleton code to implement the following two tasks:

(1) In skeleton code, it is possible that each philosopher holds a fork at his left (right) side and waits to get the other fork at his right (left) side. This may lead to a deadlock. Your **Task 1** is to avoid such deadlock

situation by revising the skeleton code as following: only one hungry philosopher at a time should be able to attempt to eat.

(2) Your **Task 2** is to revise the skeleton code so that only 4 diners (instead of 5) are allowed to attempt to grab the forks at the same time, by using condition variables in Pthread.

Both these tasks, will lead to a safe solution to the Dining Philosophers problem. In task 1 because only one philosopher picks the forks at a time, and in task 2 because as we said in class the deadlock happens only when all 5 diners attempt to get the forks at the same time.

This completes the lab.

Hand-in procedure

You will submit one zip folder with the following files: two source code files, one for each one of the tasks asked. Along with the two source code files, submit a text file where you will explain how you solved the problem by describing your solution in the code. The due is on Wednesday, Feb 21st, 11.59pm.

Grading

First task: 40pts (30 for code, and 10 for explanation), Second task: 60pts (50 for code, and 10 for explanation)