CSCI 340 - Operating Systems

Assignment 6 - Total Points 80

Objectives

In this assignment you will develop a small software application that simulates the dining philosophers problem. This assignment will allow you to gain experience, or extend your knowledge, in the following areas:

- **More 'C' Programming**: This includes variable declaration, data types, arrays, pointers, operators, expressions, selection statements, looping statements, functions, structs, and header files.
- **Thread**: Learn how to create, run, join, and terminate multiple threads.
- **Mutex**: Learn how to create, lock, and unlock a mutex.
- **Sleep**: Learn how to put a thread to sleep for a given number of nanoseconds.

System and Standard Lib Functions

In this assignment you will use the system and standard library functions listed below. Please become familiar with the syntax and usage of these calls. Detailed information about each function listed below can be found using the `man` command from the console: i.e. `man pthread_create`, will show the man page (short for manual page) for the `pthread_create` function.

- **Thread Creation**: `int pthread_create(pthread_t* thread, const pthread_attr_t* attr, void* (*start_routine) (void*), void* arg)`
- **Thread Join**: `int pthread_join(pthread_t thread, void** retval)`
- **Thread Exit**: `void pthread_exit(void* retval)`
- **Thread Kill**: `int pthread_kill(pthread_t thread, int sig)`
- **Mutex Lock**: `int pthread_mutex_lock(pthread_mutex_t* mutex)`
- **Mutex Unlock**: `int pthread_mutex_unlock(pthread_mutex_t* mutex)`

Provided Files

The three files listed below are provided to you.

- **dpsim.h**: Header file that defines the function prototypes used in this assignment. **Please note**: You may not modify or add new function definitions to this header file.
- **dpsim.c**: A file that provides the implementation of each function prototype listed in *dpsim.h*. You are to fully implement the `th_main`, `th_phil`, and `eat` functions. Please see TODO comments.
- **hw6.c**: Source code file that includes a stubbed out main function to be completed by you.

Todo

Many comments are provided above each function in the *hw6.c* and *dpsim.h* files. The provided comments give detailed instructions that are meant to guide you through this assignment. Please read them carefully and follow the instructions. For reference purposes, Fig. 1 shows which chopsticks can be used by each philosopher. For instance, philosopher one (P1) can only pickup chopstick one (c1) followed by chopstick two (c2). Your implementation must follow this chopstick ordering (i.e. right chopstick first then left chopstick) as illustrated in this diagram.

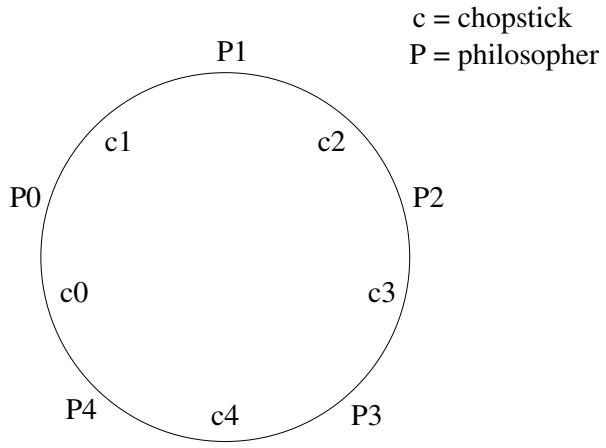

Figure 1: Table with five philosopers and five chopsticks. Each philosopher can only pickup the chopstick to his/her immediate right or left. For instance, philosopher one (P1) can only pickup chopstick one (c1) first and then chopstick two (c2)

Collaboration and Plagiarism

This is an **individual assignment**, i.e. **no collaboration is permitted**. Plagiarism will not be tolerated. Submitted solutions that are very similar (determined by the instructor) will be given a grade of zero. Please do your own work, and everything will be OK.

Submission

Create a compressed tarball, i.e. *tar.gz*, that only contains the completed *hw6.c* and *dpsim.c* files. The name of the compressed tarball must be your last name. For example, *ritchie.tar.gz* would be correct if the original co-developer of UNIX (Dennis Ritchie) submitted the assignment. Only assignments submitted in the correct format will be accepted (no exceptions). Submit the compressed tarball (via OAKS) to the Dropbox setup for this assignment. You may resubmit the compressed tarball as many times as you like, Dropbox will only keep the newest submission.

To be fair to everyone, late assignments will not be accepted. Exceptions will only be made for extenuating circumstances, i.e. death in the family, health related problems, etc. You will be given a week to complete this assignment. Poor time management is not excuse. Please do not email assignment after the due date, it will not be accepted. Please feel free to setup an appointment to discuss the assigned coding problem. I will be more than happy to listen to your approach and make suggestions. However, I cannot tell you how to code the solution. Additionally, code debugging is your job. You may use the debugger (gdb) or print statements to help understand why your solution is not working correctly, your choice.

Grading Rubric

For this assignment the grading rubric is provided in the table shown below.

Program Compiles	10 points
Program Runs with no errors	10 points
main() function implementation	10 points
th_main() function implementation	20 points
th_phil() function implementation	10 points
eat() function implementation	20 points

In particular, the assignment will be graded as follows, if the submitted solution

- does not compile: 0 of 80 points
- compiles but does not run: 10 of 80 points
- compiles and runs with errors: 15 of 80 points
- compiles and runs without errors: 20 of 80 points
- All functions correctly implemented: 80 of 80 points