Programming Assignment 2 (Hw3): Dinning Philosophers' problem

Due Date: May 1, 2018

Description

A dinning philosophers' problem is a classic synchronization problem. Though, it does not notably represent a real-world problem, it provides a significant learning value, particularly in process synchronization. It is defined in our textbook as follows:

There are 5 philosophers who spend their time just thinking and eating. They sit at the table and each has his/her own chair. There is a rice bowl in the center of the table and there are 5 chopsticks which of each is laid next to the philosopher's hand as shown below. When a philosopher thinks, he will not interact with others. Once in a while, a philosopher is hungry and tries to pick up chopsticks on his left and right-hand sides. A philosopher can pick only one chopstick at a time. When a hungry philosopher has chopsticks on both hands, he can start eating. When he is done eating, he puts both chopsticks down and starts thinking again.

Note: This picture is originally from Silberchatz's book or instructor's material.

Implement the above problem (5 philosophers) by creating 5 threads and using mutex for synchronization.

However, care must be taken to prevent a deadlock problem. One possible solution to alleviate the deadlock is known as "an asymmetric solution", that is, an odd philosopher picks up first a left chopstick and then the right one, while an even philosopher picks up first a right chopstick and then the left one.

Write your program using pthread_create for each dinning philosopher. Your program should take two arguments; a number of time to eat and a number of philosophers' eating's from the command line.

```
% dphil 10 7          // each philosopher will eat 10 times before existing. There are 7 philosophers.
Philosopher 0 is thinking...
Philosopher 1 is eating...
Philosopher 3 is thinking...
Philosopher 4 is thinking...
```

Submit your code on moodle with screenshot of your test cases.