Dining Philosopher Problem Readme

Problem: Five philosopher sit on round table and two jobs only either think or eat. Sub parts:

a) They require two forks to eat from left and right. Between every two philosophers, 1 fork is present. Therefore, total forks in table=5
b) they also require a sauce bowl for eating. Total sauce bowl present on table =2.

Since unrestricted access to fork leads to deadlock we have to prevent deadlock in this question. We prevented deadlock through two techniques.

1) Strict ordering: Even philosopher pick the right fork first and then the left fork. Odd the philosopher will pick the left fork first and then right fork. Therefore, deadlock will not happen.
2) Semaphores: Philosopher is allowed to pick only when both forks are available. Therefore preventing deadlock.

For the sauce bowl part, we just checked whether any of the bowl is available. If available, we let the philosopher use it.

Functions used:
a) Strict ordering: pthread_mutex used for just locking the fork.

b) Semaphores are used for deadlock prevention as well as locking and synchronization.