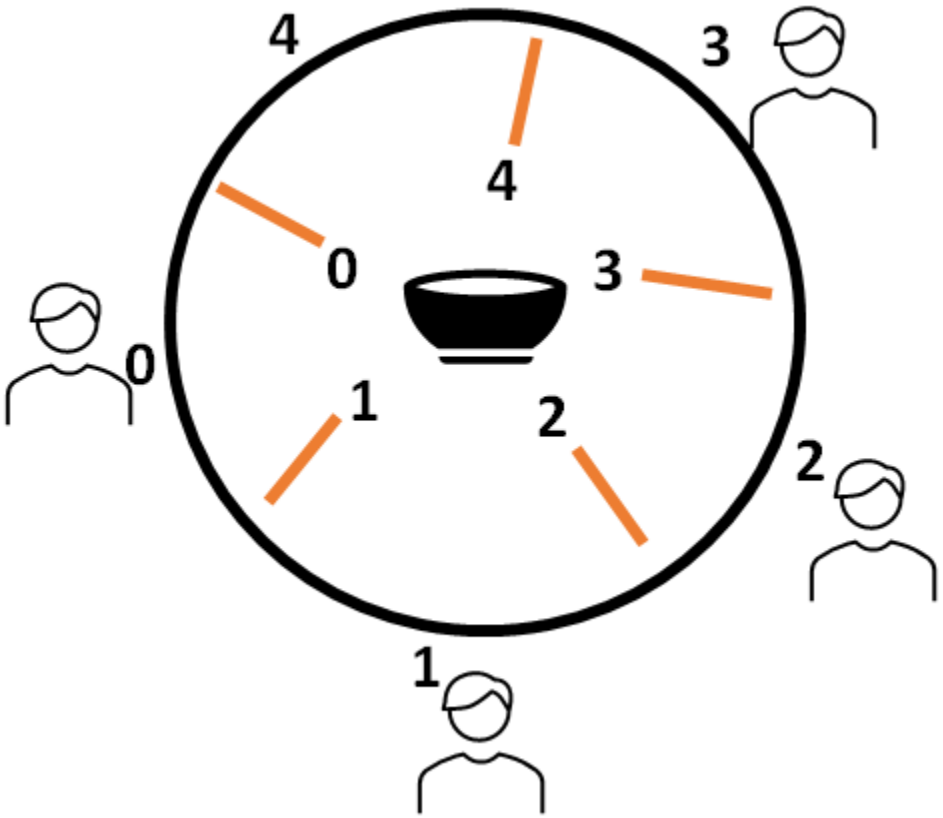
Bonus Assignment Writeup pdf



Part 1:-

Deadlock:-

part 1 is a relatively simple problem, deadlock will occur in this case only when all the thread/philosopher holds the fork which is on their left or right. In that case, all the threads will have only one fork but they need two forks to start eating.

Solution:-

To avoid deadlock we can allow all the threads to first pick the fork on their left then right except one thread which will first pick the fork on its right then on its left.

Part2:-

DeadLock situation:-

deadlock can occur when 4 of the same thread are accessing the 4 bowls and the 5th thread/philosopher is sitting ideally.

Solution:-

One possible solution to this problem would be to prevent a thread from immediately eating again after it has finished eating. We could use sleep() System call to stop the execution of that thread after it has finished eating so that other threads get a chance to take the bowl and eat. sleep(1) syscall will stop the execution of that thread for 1 second. Another solution could be to use a loop to pause the execution for 1 second. I.e

for(int i = 0 ; i<1e6 ; i++);

This will take 1 second almost to complete the loop & which would be sufficient time to switch the bowls between threads.

Part3:-

Part 3 is kind of mixture of both part 1 and part 2

Deadlock situation:-

A deadlock situation could occur in two ways when all the thread/philosopher holds the fork which is on their left or right. In that case, all the threads will have only one fork but they need two forks to start eating. second, it can occur when 4 of the same thread are accessing the 4 bowls and the 5th thread/philosopher is sitting ideally.

Solution:-

We will combine the solution of part 1 and part2 to save this program from deadlock. We will use a sleep() system call to give proper chance to all the threads to access the bowl. All threads will access the left fork first except the last thread which will access the right fork first and then the left fork.