National University of Computer and Emerging Sciences 

**Laboratory Manual**

***(Operating Systems)***

| Semester Spring 2024 |
| --- |
| Lab Instructor Mr Usman Anwer |
| Section BCS-4B |

Department of Computer Science

FAST-NU, Lahore

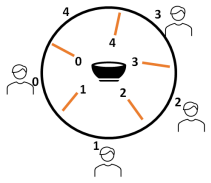
**Objective:**

To learn synchronization through Semaphores between threads

**Question # 1**

There are five philosophers sitting on a round table. Each philosopher can do two things, i) eat ii) think. Each philosopher is sitting on a chair and a bowl of rice is shared between them. The table also contains 5 chopsticks.

The scenario is illustrated by this diagram.



When a philosopher thinks, he does not interact with others. When he gets hungry, he tries to pick up the two chopsticks that are near to him, one is its left and other to its right. For example, philosopher 1 will try to pick chopsticks 1 and 2.

***Problem:***

But the philosopher can pick up only one chopstick at a time. He can not take a chopstick that is already in the hands of his neighbour. if philosopher 2 is already eating and philosopher 1 tries eating. He cannot do this, because the right chopstick is already in use and he has to wait until philosopher 2 ends eating. The philosopher starts to eat when he has both his chopsticks in his hand. After eating the philosopher puts down both the chopsticks and starts to think again.

***Hint:***

Create an array of philosophers, and one semaphore for each chopstick. The semaphores are binary. sem\_t chopstick[5]; // one semaphore for each chopstick

pthread\_t T[5] // one thread for each philosopher

*chopstick[ (i+1) % 5]* // use in signal and wait to synchronize semaphores randomly For semaphores coordination consult the solution of the previous lab attached herewith.

***Pseudocode***:

Pseudocode for a single philosopher is given.

*thread P[i]*

*while true do*

*{ THINK;*

*PICKUP(CHOPSTICK[i], CHOPSTICK[i+1 mod 5]);*

*EAT;*

*PUTDOWN(CHOPSTICK[i], CHOPSTICK[i+1 mod 5])*

*}*

**Question 2:**

We have a river that can only be crossed using a single rope, and monkeys from both sides of the river need to cross the river without causing conflicts or deadlocks. Suppose 5 monkeys are on each side waiting to cross the river.

River: Imagine a river dividing two sides, left and right.

Monkeys: There are several monkeys on each side of the river that want to cross to the other side.

Rope: There is only one rope available to cross the river. Rules:

Only one monkey can use the rope at a time.

Monkeys cannot cross in opposite directions simultaneously. Monkeys cannot cross the river without the rope.

You are required to write a program in linux C/C++ using Semaphores.

References:

1. https://www.geeksforgeeks.org/use-posix-semaphores-c/ 2. http://www.csc.villanova.edu/~mdamian/threads/posixsem.html 3.