National University of Computer and Emerging Sciences

Laboratory Manual

(Operating Systems)

Semester	Spring 2022
Lab Instructor	Saif ullah tanvir Mr Usman Anwer
Section	A

Department of Computer Science

FAST-NU, Lahore

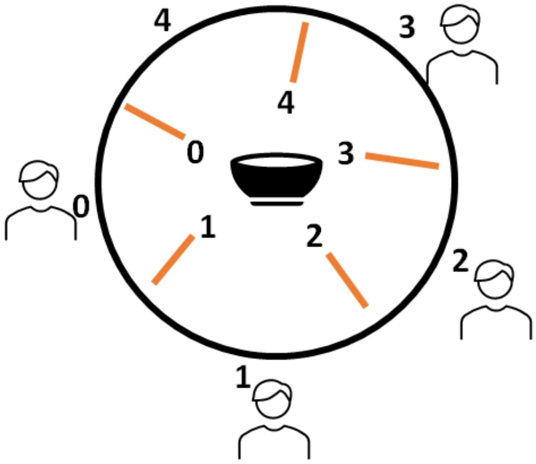
Objective:

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])
   }
```

Output: The output should be similar to this. (with 5 philosophers)

```
Philosopher 1 wants to eat
Philosopher 3 wants to eat
Philosopher 0 wants to eat
Philosopher 2 wants to eat
Philosopher 2 tries to pick left chopstick
Philosopher 2 picks the left chopstick
Philosopher 2 tries to pick the right chopstick
Philosopher 2 picks the right chopstick
Philosopher 2 begins to eat
Philosopher 1 tries to pick left chopstick
Philosopher 1 picks the left chopstick
Philosopher 1 tries to pick the right chopstick
Philosopher 0 tries to pick left chopstick
Philosopher 0 picks the left chopstick
Philosopher 0 tries to pick the right chopstick
Philosopher 3 tries to pick left chopstick
Philosopher 4 wants to eat
Philosopher 4 tries to pick left chopstick
Philosopher 4 picks the left chopstick
Philosopher 4 tries to pick the right chopstick
Philosopher 2 has finished eating
Philosopher 2 leaves the right chopstick
Philosopher 2 leaves the left chopstick
Philosopher 3 picks the left chopstick
Philosopher 3 tries to pick the right chopstick
Philosopher 1 picks the right chopstick
Philosopher 1 begins to eat
Philosopher 1 has finished eating
Philosopher 1 leaves the right chopstick
Philosopher 1 leaves the left chopstick
Philosopher 0 picks the right chopstick
Philosopher 0 begins to eat
Philosopher 0 has finished eating
Philosopher 0 leaves the right chopstick
Philosopher 0 leaves the left chopstick
Philosopher 4 picks the right chopstick
Philosopher 4 begins to eat
Philosopher 4 has finished eating
Philosopher 4 leaves the right chopstick
Philosopher 4 leaves the left chopstick
Philosopher 3 picks the right chopstick
```