STEVENS
INSTITUTE of TECHNOLOGY
THE INNOVATION UNIVERSITY

CS 492: Operating Systems

Inter Process Communication Programming – Pre-Lab Session

Instructor: Iraklis Tsekourakis

Email: itsekour@stevens.edu

Lab Description

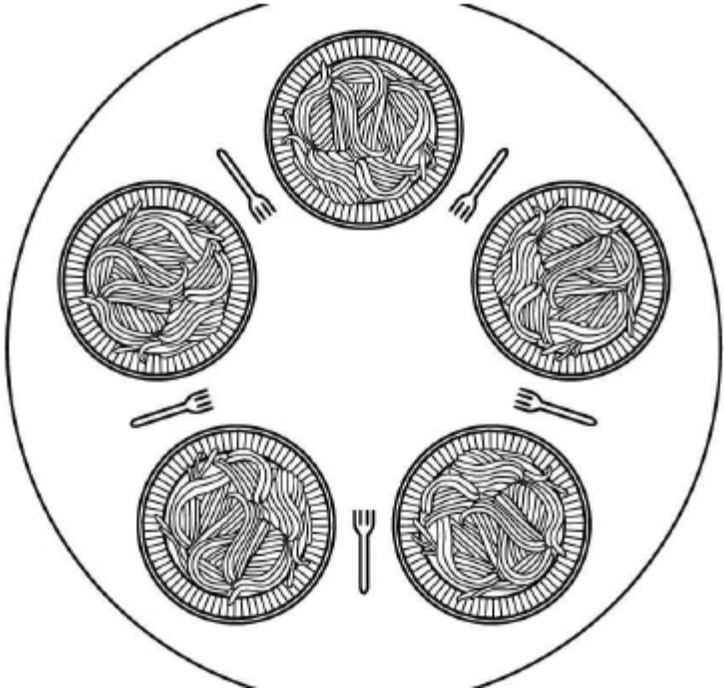
- Friday (Feb 16) in class
- Bring your laptop to class
- Task: Implementation of *Dining Philosophers Problem* by using pthread programming (**C**, C++)

Today

- Pre-lab session
  - Dining philosopher problem
  - Overview of skeleton code for lab

Dining Philosopher Problem

- A typical IPC problem
  - Philosophers eat/think
  - Eating needs 2 forks
  - Pick one fork at a time



Implementing Dining Philosophers Problem using Semphores (Our skeleton code)

```
#define N 5                                     /* number of philosophers */

void philosopher(int i)                         /* i: philosopher number, from 0 to 4 */
{
    while (TRUE) {
        think( );                               /* philosopher is thinking */
        take_fork(i);                           /* take left fork */
        take_fork((i+1) % N);                   /* take right fork; % is modulo operator */
        eat( );                                 /* yum-yum, spaghetti */
        put_fork(i);                            /* put left fork back on the table */
        put_fork((i+1) % N);                    /* put right fork back on the table */
    }
}
```

Our Lab Session

- The above solution is provided as skeleton code.
- The URL of skeleton code is in the lab description
- Download the skeleton code before the lab
- The problem of the skeleton code
  - Possible deadlock!

In Our Lab

- Two tasks
  - **Task 1**: avoid deadlock by revising the skeleton code as following: only one hungry philosopher at a time should be able to attempt to eat.
  - **Task 2**: to revise the skeleton code so that only 4 diners are allowed to eat at a time, using condition variables in pthread.

Policy

- You will work **with your partner**
- The skeleton code was implemented in C. You are welcome to change it into C++, if you can finish the lab in time.
- You may need to log into a CS Linux machine for compiling and running the code.

Hand-in Procedure

- Ideally you will complete the code in class, but you will have until Sunday, 11.59pm to submit on Canvas.
- In your submission, include a short text file, with explanation of your code, and how you address both tasks.

Grading Scheme

Task1 (40%)		**Task2 (60%)**	
Coding(30%)	Student explanation (10%)	Coding(50%)	Student explanation(10%)