**Programming Assignment #3**

1. **Introduction**

* **What you will talk about/do**

I will write a multi-threaded program, in Java, that implements the well-known Dining Philosophers problem. This particular problem is often used to demonstrate the use of mutual exclusion algorithms and deadlock prevention in multi-threaded applications.

* **Overview of the rest of your report**

The scenario is that there are some number N of oriental philosophers sitting around a round table (for our implementation we will use N=5). Each philosopher has one chopstick at his right hand and one chopstick at his left hand. However, the right hand chopstick for philosopher n is the left hand chopstick for philosopher (n+1) mod N. In other words, there are N chopsticks for N philosophers (one between each pair of philosophers).

* **Project**
  + **Your approach to the problem**

Use Java threads for each philosopher. Use Java synchronize to lock the chopsticks. Use the Java wait and notify methods to implement the condition variables.

* + **Design**
    - Dining class

Text

Description automatically generated

Create thread == philosopher

To prevent logresults, using while loop.

* + - Class Philosopher

Text

Description automatically generated

First randomdelay means that Philosopher is thinking

Then using monitor, philosopher can eat.

After eating, philosopher, putdown chopsticks and return thinking state.

If Total Eats exceed the MaxEats(==500), then break the loop, and finish the program.

* + - Class PhilosopherMonitor

Text

Description automatically generated

In this monitor, check philosopher can eat or not.

When pickup, they are can eat. But must check, their neighbor is eating or not.

After eating, they put down chopsticks. And notifyall.

* + Results

A picture containing table

Description automatically generated

1. **Conclusions**

I try to finish program appropriately. For example, I’m using while loop condition setting TotalEats don’t exceed MaxEats. But it does not work well.

So, the result of my program almost print out “Error! Eating more than MAZ.. exiting..”.

It is regrettable that it cannot produce perfect results. So, I will look for more ways to eliminate this error.