Marisol Morales

Student ID: 029984979

CECS 326 Sec 02

Professor Xu

Project 2 “Synchronization: Dining Philosopher problem” Report

For this project I decided to do it in Python, although I may have used threads and time in C++ I haven’t in python yet. This is when I decided to settle on the time and threading library, there is also the random library which is used later on in our code. The thread library allows us to utilize mutexes and semaphores within our code which is a key component for synchronization, we use this for our forks, mutex lock, and threads. This is required as it will make sure that one fork is only able to be held by one philosopher when we run, and for the mutex lock it allows them to basically pick them up in a controlled manner to prevent possible deadlock and starvation to occur.

We have three functions that we use in our code, def pickup_fork(index), def return_forks(index), and our def philosopher(index). In the def pickup_fork(index) function that is when we set the code up for when our philosopher wants to start eating they can pick up their forks, since we have two forks I had to think of an optimal way to assign the forks without causing issues, that is why i settled on “left_fork = index right_fork = (index + 1) % num_forks” the philosopher will get assigned to the fork to their left and to their right, and it will continue in this this cycle, def return_forks(index) focuses on letting the philosopher return their forks to let the others pick those up and eat, and def philosopher(index) is where we use our two functions to begin the stimulation of eating, thinking, and returning our forks. For thinking that is where we use our random library as we randomly select our philosopher to wait 1 to 3 seconds before

continuing to the next step. Finally we create our philosopher threads, start those threads with a for loop, and join those threads.

YouTube Link: https://youtu.be/XUPszMFgVQY?si=cQdxbewhois1h3b5