CSE 4502 (SWE) [Operating Systems Lab]
Lab # 04_2

Lab Tasks:

1. The Dinning Philosopher Problem. Five silent philosophers sit at a round table with bowls of rice. Chopsticks are placed between each pair of adjacent philosophers. Each philosopher must alternately think and eat. However, a philosopher can only eat rice when they have both left and right chopsticks. Each chopstick can be held by only one philosopher and so a philosopher can use the chopstick only if it is not being used by another philosopher. After an individual philosopher finishes eating, they need to put down both chopsticks so that the chopsticks become available to others. A philosopher can only take the chopstick on their right or the one on their left as they become available and they cannot start eating before getting both chopsticks.
   Eating is not limited by the remaining amounts of rice or stomach space; an infinite supply and an infinite demand are assumed.
   The dining philosopher is a classic synchronization problem. One of the solutions to this problem is to allow at best four philosophers to eat at a time. If four philosophers are already eating, the fifth must wait until at least one philosopher finishes eating.
   Write a semaphore-based solution to the Dining Philosopher Problem that will allow only four philosophers to eat simultaneously.

2. Task 2 from Lab#04_1 requires the parent thread to wait for the child thread to finish its execution before printing out the computed values. If we let the parent thread access the Fibonacci numbers as soon as they have been computed by the child thread—rather than waiting for the child thread to terminate—what changes would be necessary to the solution for this task? Implement your modified solution.

3. The Sleeping-Barber Problem. A barbershop consists of a waiting room with n chairs and a barber room with one barber chair. If there are no customers to be served, the barber goes to sleep. If a customer enters the barbershop and all chairs are occupied, then the customer leaves the shop. If the barber is busy but chairs are available, then the customer sits in one of the free chairs. If the barber is asleep, the customer wakes up the barber. Write a program to coordinate the barber and the customers.