WEEK 7

By: Priyanshu Singh (RA2311026010414)

Program:

```
#include<stdio.h>
#include<stdlib.h>
#include<pthread.h>
#include<semaphore.h>
#include<unistd.h>

sem_t room;
sem_t chopstick[5];

void * philosopher(void * num) {
   int phil = *(int *)num;

   // Philosopher attempts to enter the room
   sem_wait(&room);
   printf("\nPhilosopher %d has entered the room.", phil);

   // Philosopher picks up the chopsticks
   sem_wait(&chopstick[phil]);          // Picking up left chopstick
   sem_wait(&chopstick[(phil + 1) % 5]);  // Picking up right chopstick

   // Philosopher eats
```

```
    eat(phil);
    sleep(2);  // Simulate eating time

    // Philosopher puts down the chopsticks
    sem_post(&chopstick[(phil + 1) % 5]);  // Putting down right chopstick
    sem_post(&chopstick[phil]);            // Putting down left chopstick

    // Philosopher leaves the room
    printf("\nPhilosopher %d has finished eating and left the room.", phil);
    sem_post(&room);

    return NULL;
}

void eat(int phil) {
    printf("\nPhilosopher %d is eating.", phil);
}

int main() {
    int i, a[5];
    pthread_t tid[5];
```

```
    // Initialize the room semaphore to 4 (allowing only 4 philosophers in the room at a time)
    sem_init(&room, 0, 4);

    // Initialize the chopstick semaphores (one for each philosopher)
    for(i = 0; i < 5; i++) {
        sem_init(&chopstick[i], 0, 1);
    }

    // Create philosopher threads
    for(i = 0; i < 5; i++) {
        a[i] = i;
        pthread_create(&tid[i], NULL, philosopher, (void *)&a[i]);
    }

    // Wait for all philosopher threads to finish
    for(i = 0; i < 5; i++) {
        pthread_join(tid[i], NULL);
    }

    return 0;
}
```

Output:

Philosopher 0 has entered the room.

Philosopher 0 is eating.

Philosopher 1 has entered the room.

Philosopher 2 has entered the room.

Philosopher 3 has entered the room.

Philosopher 0 has finished eating and left the room.

Philosopher 1 is eating.

Philosopher 4 has entered the room.

Philosopher 1 has finished eating and left the room.

Philosopher 2 is eating.

Philosopher 2 has finished eating and left the room.

Philosopher 3 is eating.

Philosopher 3 has finished eating and left the room.

Philosopher 4 is eating.

Philosopher 4 has finished eating and left the room.