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

#define N 2
sem_t forks[N];
pthread_mutex_t mutex;

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

  while (1) {
    printf("Philosopher %d is thinking...\n", id);
    sleep(1);

    pthread_mutex_lock(&mutex);
    sem_wait(&forks[id]);
    sem_wait(&forks[(id + 1) % N]);
    pthread_mutex_unlock(&mutex);

    printf("Philosopher %d is eating...\n", id);
    sleep(2);

    sem_post(&forks[id]);
    sem_post(&forks[(id + 1) % N]);

    printf("Philosopher %d finished eating and is thinking again...\n", id);
  }
}

int main() {
  pthread_t philosophers[N];
  int ids[N];
  pthread_mutex_init(&mutex, NULL);

  for (int i = 0; i < N; i++) {
    sem_init(&forks[i], 0, 1);
    ids[i] = i;
  }

  for (int i = 0; i < N; i++) {
    pthread_create(&philosophers[i], NULL, philosopher, &ids[i]);
  }

  for (int i = 0; i < N; i++) {
    pthread_join(philosophers[i], NULL);
  }

  for (int i = 0; i < N; i++) {
    sem_destroy(&forks[i]);
  }
  pthread_mutex_destroy(&mutex);
  return 0;
}
```