Dining-Philosophers problem

#include <stdio.h>

#include <pthread.h>

#include <semaphore.h>

#include <unistd.h>

#define N 5

sem\_t forks[N];

pthread\_t philosophers[N];

void\* philosopher(void\* num) {

int id = *(int*)num;

int left = id;

int right = (id + 1) % N;

while (1) {  
 printf("Philosopher %d is thinking.\n", id);  
 sleep(1);  
 sem\_wait(&forks[left]);  
 sem\_wait(&forks[right]);  
 printf("Philosopher %d is eating.\n", id);  
 sleep(2);  
 sem\_post(&forks[left]);  
 sem\_post(&forks[right]);  
 printf("Philosopher %d finished eating.\n", id);  
 sleep(1);  
}  
return NULL;

}

int main() { int i; int ids[N];

// Initialize semaphores  
for (i = 0; i < N; i++) {  
 sem\_init(&forks[i], 0, 1);  
}  
  
// Create philosopher threads  
for (i = 0; i < N; i++) {  
 ids[i] = i;  
 pthread\_create(&philosophers[i], NULL, philosopher, &ids[i]);  
}  
  
// Join threads (infinite loop in simulation)  
for (i = 0; i < N; i++) {  
 pthread\_join(philosophers[i], NULL);  
}  
  
return 0;

}