21.DPP

#include <pthread.h>

#include <semaphore.h>

#include <stdio.h>

#include <unistd.h>

#define N 5

#define LEFT (i + N - 1) % N

#define RIGHT (i + 1) % N

#define THINKING 0

#define HUNGRY 1

#define EATING 2

int state[N];

sem\_t mutex;

sem\_t s[N];

void test(int i) {

if (state[i] == HUNGRY && state[LEFT] != EATING && state[RIGHT] != EATING) {

state[i] = EATING;

sem\_post(&s[i]);

}

}

void take\_forks(int i) {

sem\_wait(&mutex);

state[i] = HUNGRY;

test(i);

sem\_post(&mutex);

sem\_wait(&s[i]);

}

void put\_forks(int i) {

sem\_wait(&mutex);

state[i] = THINKING;

test(LEFT);

test(RIGHT);

sem\_post(&mutex);

}

void \*philosopher(void \*num) {

int i = \*(int\*)num;

while (1) {

printf("Philosopher %d is thinking\n", i + 1);

sleep(1);

take\_forks(i);

printf("Philosopher %d is eating\n", i + 1);

sleep(1);

put\_forks(i);

}

}

int main() {

int i;

pthread\_t thread\_id[N];

int num[N];

sem\_init(&mutex, 0, 1);

for (i = 0; i < N; i++) {

sem\_init(&s[i], 0, 0);

}

for (i = 0; i < N; i++) {

num[i] = i;

pthread\_create(&thread\_id[i], NULL, philosopher, &num[i]);

}

for (i = 0; i < N; i++) {

pthread\_join(thread\_id[i], NULL);

}

}

OUTPUT

