//dining philosopher

#include <pthread.h>

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#define NUM\_PHILOSOPHERS 5

#define THINKING 0

#define HUNGRY 1

#define EATING 2

pthread\_mutex\_t mutex;

pthread\_cond\_t condition[NUM\_PHILOSOPHERS];

int state[NUM\_PHILOSOPHERS];

void test(int i)

{

if (state[i] == HUNGRY &&

state[(i + 4) % 5] != EATING &&

state[(i + 1) % 5] != EATING)

{

state[i] = EATING;

printf("Philosopher %d is Eating.\n", i);

pthread\_cond\_signal(&condition[i]);

}

}

void pickup\_forks(int philosopher\_number) {

pthread\_mutex\_lock(&mutex);

state[philosopher\_number] = HUNGRY;

printf("Philosopher %d is Hungry.\n", philosopher\_number);

test(philosopher\_number);

while (state[philosopher\_number] != EATING) {

pthread\_cond\_wait(&condition[philosopher\_number], &mutex);

}

pthread\_mutex\_unlock(&mutex);

}

void return\_forks(int philosopher\_number) {

pthread\_mutex\_lock(&mutex);

state[philosopher\_number] = THINKING;

printf("Philosopher %d is Thinking.\n", philosopher\_number);

test((philosopher\_number + 4) % 5);

test((philosopher\_number + 1) % 5);

pthread\_mutex\_unlock(&mutex);

}

void \*philosopher(void \*arg)

{

int philosopher\_number = \*(int \*)arg;

while (1)

{

sleep(rand() % 3 + 1);// Hungry, want to eat

pickup\_forks(philosopher\_number);// Eating

sleep(rand() % 3 + 1);// Done eating, release forks

return\_forks(philosopher\_number);

}

}

int main()

{

pthread\_t philosophers[NUM\_PHILOSOPHERS];

int philosopher\_numbers[NUM\_PHILOSOPHERS];

pthread\_mutex\_init(&mutex, NULL);

for (int i = 0; i < NUM\_PHILOSOPHERS; ++i)

{

pthread\_cond\_init(&condition[i], NULL);

philosopher\_numbers[i] = i;

pthread\_create(&philosophers[i], NULL, philosopher, (void \*)&philosopher\_numbers[i]);

}

for (int i = 0; i < NUM\_PHILOSOPHERS; ++i)

{

pthread\_join(philosophers[i], NULL);

}

pthread\_mutex\_destroy(&mutex);

for (int i = 0; i < NUM\_PHILOSOPHERS; ++i)

{

pthread\_cond\_destroy(&condition[i]);

}

return 0;

}

Output:

