### Write a C program to simulate Earliest deadline first scheduling.

**PROGRAM:**

#include <stdio.h>

#include <stdlib.h>

#include <pthread.h>

#include <semaphore.h>

#include <unistd.h>

#define N 5

enum { THINKING, HUNGRY, EATING };

int state[N];

sem\_t mutex;

sem\_t sem[N];

void test(int i) {

int left = (i + N - 1) % N;

int right = (i + 1) % N;

if (state[i] == HUNGRY &&

state[left] != EATING &&

state[right] != EATING) {

state[i] = EATING;

sem\_post(&sem[i]);

}

}

void take\_forks(int i) {

sem\_wait(&mutex);

state[i] = HUNGRY;

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

test(i);

sem\_post(&mutex);

sem\_wait(&sem[i]);

}

void put\_forks(int i) {

sem\_wait(&mutex);

state[i] = THINKING;

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

test((i + N - 1) % N);

test((i + 1) % N);

sem\_post(&mutex);

}

void\* philosopher(void\* num) {

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

while (1) {

sleep(rand() % 3 + 1);

take\_forks(id);

printf("Philosopher %d is EATING\n", id);

sleep(rand() % 2 + 1);

put\_forks(id);

}

}

int main() {

pthread\_t tid[N];

int ids[N];

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

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

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

state[i] = THINKING;

ids[i] = i;

}

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

pthread\_create(&tid[i], NULL, philosopher, &ids[i]);

}

sleep(20);

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

pthread\_cancel(tid[i]);

sem\_destroy(&sem[i]);

}

sem\_destroy(&mutex);

return 0;

}

**OUTPUT :**