|  |
| --- |
| #include <pthread.h> |
|  | #include <stdio.h> |
|  | #include <unistd.h> |
|  | #include <time.h> |
|  | #include <stdlib.h> |
|  |  |
|  | #define NUMBER\_OF\_PHILOSOPHERS 5 |
|  |  |
|  | void \*philosopher(void \*); |
|  | void think(int); |
|  | void pickUp(int); |
|  | void eat(int); |
|  | void putDown(int); |
|  |  |
|  | pthread\_mutex\_t chopsticks[NUMBER\_OF\_PHILOSOPHERS]; |
|  | pthread\_t philosophers[NUMBER\_OF\_PHILOSOPHERS]; |
|  | pthread\_attr\_t attributes[NUMBER\_OF\_PHILOSOPHERS]; |
|  |  |
|  | int main() { |
|  | int i; |
|  | srand(time(NULL)); |
|  | for (i = 0; i < NUMBER\_OF\_PHILOSOPHERS; ++i) { |
|  | pthread\_mutex\_init(&chopsticks[i], NULL); |
|  | } |
|  |  |
|  | for (i = 0; i < NUMBER\_OF\_PHILOSOPHERS; ++i) { |
|  | pthread\_attr\_init(&attributes[i]); |
|  | } |
|  |  |
|  | for (i = 0; i < NUMBER\_OF\_PHILOSOPHERS; ++i) { |
|  | pthread\_create(&philosophers[i], &attributes[i], philosopher, (void \*)(i)); |
|  | } |
|  |  |
|  | for (i = 0; i < NUMBER\_OF\_PHILOSOPHERS; ++i) { |
|  | pthread\_join(philosophers[i], NULL); |
|  | } |
|  | return 0; |
|  | } |
|  |  |
|  | void \*philosopher(void \*philosopherNumber) { |
|  | while (1) { |
|  | think(philosopherNumber); |
|  | pickUp(philosopherNumber); |
|  | eat(philosopherNumber); |
|  | putDown(philosopherNumber); |
|  | } |
|  | } |
|  |  |
|  | void think(int philosopherNumber) { |
|  | int sleepTime = rand() % 3 + 1; |
|  | printf("Philosopher %d will think for %d seconds\n", philosopherNumber, sleepTime); |
|  | sleep(sleepTime); |
|  | } |
|  |  |
|  | void pickUp(int philosopherNumber) { |
|  | int right = (philosopherNumber + 1) % NUMBER\_OF\_PHILOSOPHERS; |
|  | int left = (philosopherNumber + NUMBER\_OF\_PHILOSOPHERS) % NUMBER\_OF\_PHILOSOPHERS; |
|  | if (philosopherNumber & 1) { |
|  | printf("Philosopher %d is waiting to pick up chopstick %d\n", philosopherNumber, right); |
|  | pthread\_mutex\_lock(&chopsticks[right]); |
|  | printf("Philosopher %d picked up chopstick %d\n", philosopherNumber, right); |
|  | printf("Philosopher %d is waiting to pick up chopstick %d\n", philosopherNumber, left); |
|  | pthread\_mutex\_lock(&chopsticks[left]); |
|  | printf("Philosopher %d picked up chopstick %d\n", philosopherNumber, left); |
|  | } |
|  | else { |
|  | printf("Philosopher %d is waiting to pick up chopstick %d\n", philosopherNumber, left); |
|  | pthread\_mutex\_lock(&chopsticks[left]); |
|  | printf("Philosopher %d picked up chopstick %d\n", philosopherNumber, left); |
|  | printf("Philosopher %d is waiting to pick up chopstick %d\n", philosopherNumber, right); |
|  | pthread\_mutex\_lock(&chopsticks[right]); |
|  | printf("Philosopher %d picked up chopstick %d\n", philosopherNumber, right); |
|  | } |
|  | } |
|  |  |
|  | void eat(int philosopherNumber) { |
|  | int eatTime = rand() % 3 + 1; |
|  | printf("Philosopher %d will eat for %d seconds\n", philosopherNumber, eatTime); |
|  | sleep(eatTime); |
|  | } |
|  |  |
|  | void putDown(int philosopherNumber) { |
|  | printf("Philosopher %d will will put down her chopsticks\n", philosopherNumber); |
|  | pthread\_mutex\_unlock(&chopsticks[(philosopherNumber + 1) % NUMBER\_OF\_PHILOSOPHERS]); |
|  | pthread\_mutex\_unlock(&chopsticks[(philosopherNumber + NUMBER\_OF\_PHILOSOPHERS) % NUMBER\_OF\_PHILOSOPHERS]); |
|  | } |