Pseudo code cho bài toán bữa ăn của các triết gia

- Giả sử có 5 triết gia ngồi quanh một bàn tròn, mỗi người có một đĩa và một nĩa. Để ăn, một triết gia cần hai nĩa. Mỗi triết gia có thể đang suy nghĩ hoặc đang ăn. Để ăn, một triết gia cần lấy hai nĩa bên cạnh mình. Nếu hai nĩa đều có sẵn, triết gia sẽ ăn trong một thời gian ngắn rồi đặt nĩa xuống và tiếp tục suy nghĩ. Nếu một triết gia đói, nhưng hai nĩa bên cạnh không có sẵn, thì triết gia đó sẽ đợi cho đến khi có đủ hai nĩa để ăn. Mỗi triết gia có thể lấy nĩa bên trái hoặc bên phải của mình, nhưng không thể lấy nĩa của người bên cạnh. Mỗi triết gia có thể đặt nĩa xuống bất cứ lúc nào. Mục tiêu là thiết kế một giải pháp để ai cũng có thể được ăn, suy nghĩ.

```
#define N 5     // Số triết gia

#define LEFT (i + N - 1) % N // Số thứ tự của triết gia bên trái
#define RIGHT (i + 1) % N // Số thứ tự của triết gia bên phải

#define THINKING 0  // Trạng thái suy nghĩ
#define HUNGRY 1   // Trạng thái đói, muốn lấy nĩa
#define EATING 2  // Trạng thái ăn

int state[N];   // Mảng lưu trạng thái của các triết gia
sem_t mutex;   // Semaphore để đảm bảo chỉ một triết gia thay đổi trạng thái tại
một thời điểm
sem_t s[N];  // Semaphore để đảm bảo triết gia có thể ăn

void think() {
    // Triết gia đang suy nghĩ
    sleep(1);
}

void eat() {
    // Triết gia đang ăn
    sleep(3);
}

void philosopher(int i) {
    while (true) {
        think();    // Triết gia suy nghĩ
        take_forks(i); // Triết gia lấy nĩa
        eat(); // Triết gia ăn
        put_forks(i); // Triết gia đặt nĩa xuống
    }
}

void take_forks(int i) {
    sem_wait(&mutex); // Đảm bảo chỉ một triết gia thay đổi trạng thái tại một
thời điểm
    state[i] = HUNGRY; // Triết gia đói
    test(i); // Thử lấy nĩa
    sem_post(&mutex);
    sem_wait(&s[i]); // Nếu không lấy được nĩa, triết gia đợi
}
```

```
void put_forks(int i) {
    sem_wait(&mutex); // Đảm bảo chỉ một triết gia thay đổi trạng thái tại một
thời điểm
    state[i] = THINKING; // Triết gia suy nghĩ
    test(LEFT); // Kiểm tra xem người bên trái có thể ăn không
    test(RIGHT); // Kiểm tra xem người bên phải có thể ăn không
    sem_post(&mutex);
}

void test(int i) {
    if (state[i] == HUNGRY && state[LEFT] != EATING && state[RIGHT] != EATING) {
        // Nếu triết gia đói và hai triết gia bên cạnh không ăn, thì triết gia có
thể bắt đầu ăn
        state[i] = EATING;
        sem_post(&s[i]); // Báo hiệu là triết gia có thể bắt đầu ăn
    }
}

int main() {
    // Khởi tạo semaphores
    sem_init(&mutex, 0, 1);
    for (int i = 0; i < N; ++i) {
        sem_init(&s[i], 0, 0);
    }


    // ...
    // (Chưa giải phóng các semaphores và không có điều kiện dừng)
    // ...


    return 22;
}
```