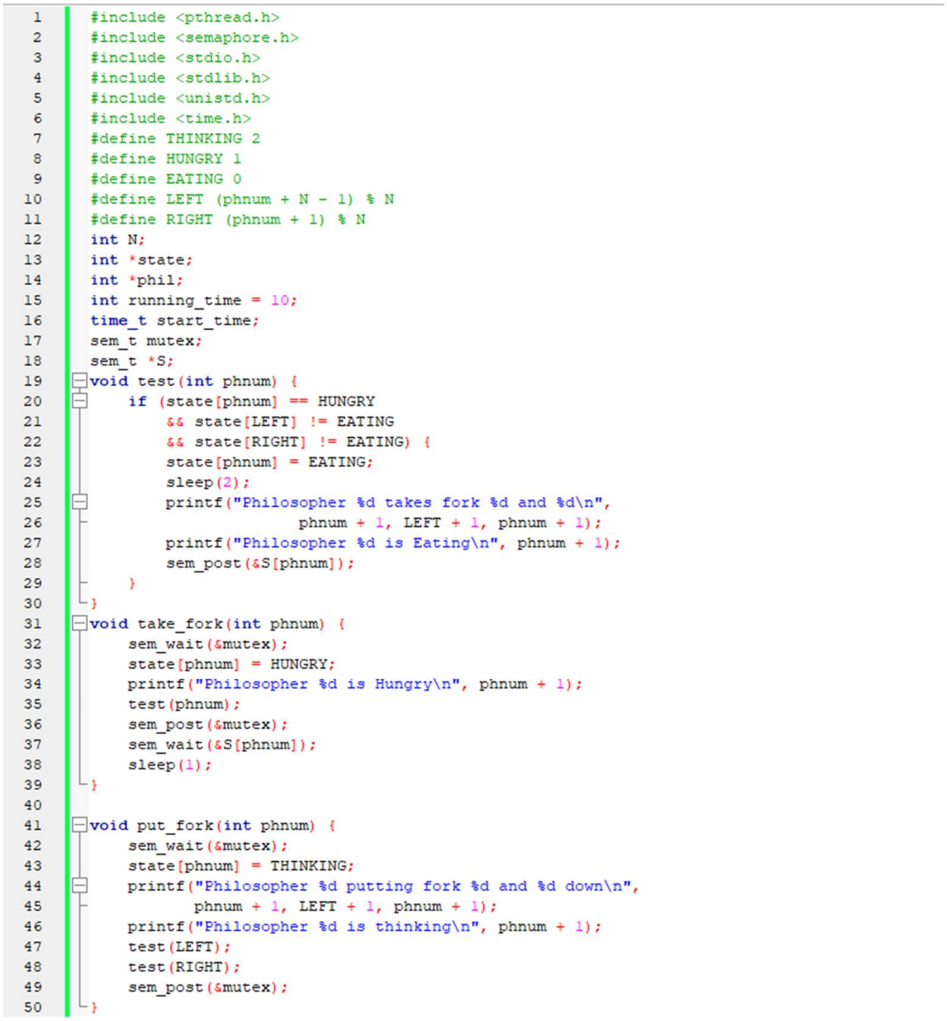
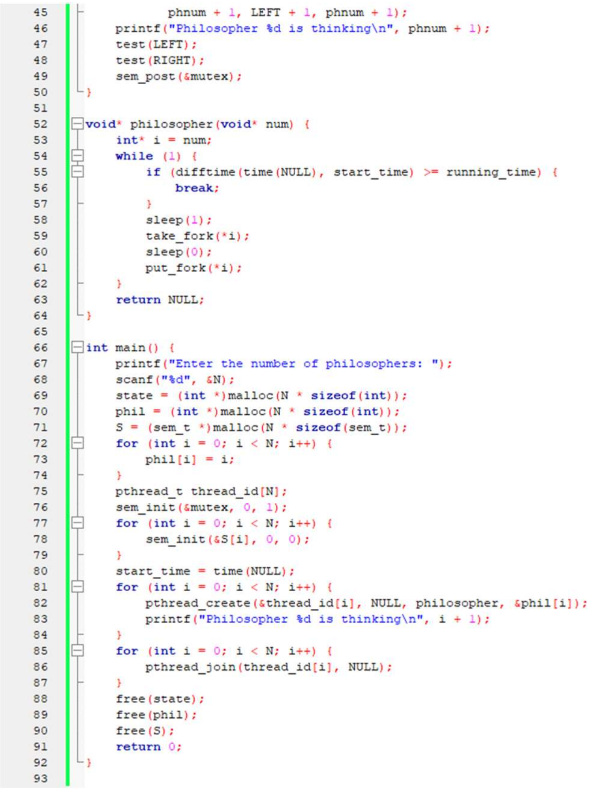
Write a C program to simulate the concept of Dining-Philosophers problem.

INPUT:



```c
#include <pthread.h>
#include <semaphore.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <time.h>
#define THINKING 2
#define HUNGRY 1
#define EATING 0
#define LEFT (phnum + N - 1) % N
#define RIGHT (phnum + 1) % N
int N;
int *state;
int *phil;
int running_time = 10;
time_t start_time;
sem_t mutex;
sem_t *S;
void test(int phnum) {
    if (state[phnum] == HUNGRY
        && state[LEFT] != EATING
        && state[RIGHT] != EATING) {
        state[phnum] = EATING;
        sleep(2);
        printf("Philosopher %d takes fork %d and %d\n",
                      phnum + 1, LEFT + 1, phnum + 1);
        printf("Philosopher %d is Eating\n", phnum + 1);
        sem_post(&S[phnum]);
    }
}
void take_fork(int phnum) {
    sem_wait(&mutex);
    state[phnum] = HUNGRY;
    printf("Philosopher %d is Hungry\n", phnum + 1);
    test(phnum);
    sem_post(&mutex);
    sem_wait(&S[phnum]);
    sleep(1);
}

void put_fork(int phnum) {
    sem_wait(&mutex);
    state[phnum] = THINKING;
    printf("Philosopher %d putting fork %d and %d down\n",
           phnum + 1, LEFT + 1, phnum + 1);
    printf("Philosopher %d is thinking\n", phnum + 1);
    test(LEFT);
    test(RIGHT);
    sem_post(&mutex);
}
```

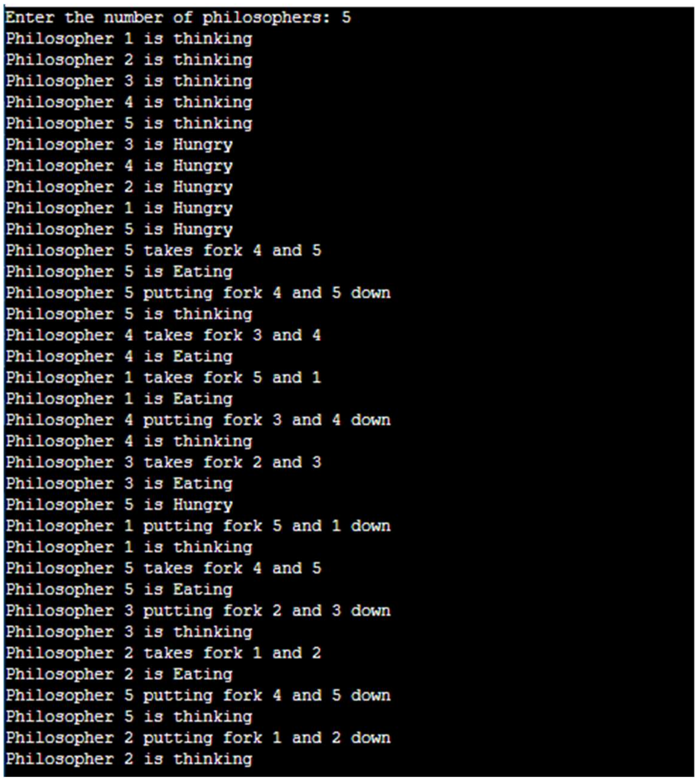


```c
            phnum + 1, LEFT + 1, phnum + 1);
    printf("Philosopher %d is thinking\n", phnum + 1);
    test(LEFT);
    test(RIGHT);
    sem_post(&mutex);
}

void* philosopher(void* num) {
    int* i = num;
    while (1) {
        if (difftime(time(NULL), start_time) >= running_time) {
            break;
        }
        sleep(1);
        take_fork(*i);
        sleep(0);
        put_fork(*i);
    }
    return NULL;
}

int main() {
    printf("Enter the number of philosophers: ");
    scanf("%d", &N);
    state = (int *)malloc(N * sizeof(int));
    phil = (int *)malloc(N * sizeof(int));
    S = (sem_t *)malloc(N * sizeof(sem_t));
    for (int i = 0; i < N; i++) {
        phil[i] = i;
    }
    pthread_t thread_id[N];
    sem_init(&mutex, 0, 1);
    for (int i = 0; i < N; i++) {
        sem_init(&S[i], 0, 0);
    }
    start_time = time(NULL);
    for (int i = 0; i < N; i++) {
        pthread_create(&thread_id[i], NULL, philosopher, &phil[i]);
        printf("Philosopher %d is thinking\n", i + 1);
    }
    for (int i = 0; i < N; i++) {
        pthread_join(thread_id[i], NULL);
    }
    free(state);
    free(phil);
    free(S);
    return 0;
}
```

OUTPUT:



```
Enter the number of philosophers: 5
Philosopher 1 is thinking
Philosopher 2 is thinking
Philosopher 3 is thinking
Philosopher 4 is thinking
Philosopher 5 is thinking
Philosopher 3 is Hungry
Philosopher 4 is Hungry
Philosopher 2 is Hungry
Philosopher 1 is Hungry
Philosopher 5 is Hungry
Philosopher 5 takes fork 4 and 5
Philosopher 5 is Eating
Philosopher 5 putting fork 4 and 5 down
Philosopher 5 is thinking
Philosopher 4 takes fork 3 and 4
Philosopher 4 is Eating
Philosopher 1 takes fork 5 and 1
Philosopher 1 is Eating
Philosopher 4 putting fork 3 and 4 down
Philosopher 4 is thinking
Philosopher 3 takes fork 2 and 3
Philosopher 3 is Eating
Philosopher 5 is Hungry
Philosopher 1 putting fork 5 and 1 down
Philosopher 1 is thinking
Philosopher 5 takes fork 4 and 5
Philosopher 5 is Eating
Philosopher 3 putting fork 2 and 3 down
Philosopher 3 is thinking
Philosopher 2 takes fork 1 and 2
Philosopher 2 is Eating
Philosopher 5 putting fork 4 and 5 down
Philosopher 5 is thinking
Philosopher 2 putting fork 1 and 2 down
Philosopher 2 is thinking
```