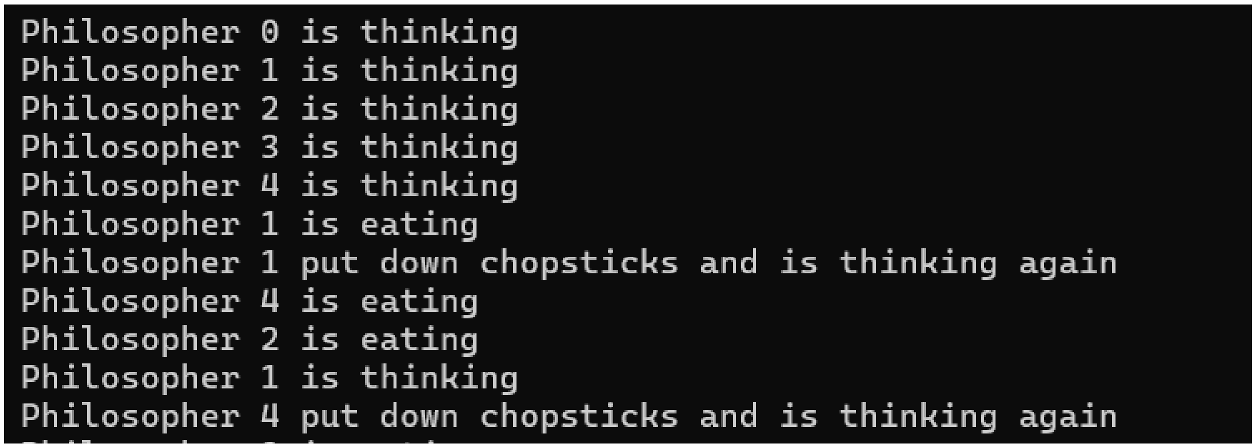
OS - DINING PHILOSOPHERS PROBLEM

4. Write a C program to simulate:
b) Dining-Philosophers problem using semaphores.

Code:

```
#include <stdio.h>
#include <pthread.h>
#include <unistd.h>

#define N 5 // number of philosophers

int chopstick[N] = {1, 1, 1, 1, 1}; // 1 means available
int mutex = 1;

int wait(int *s) {
    while (*s <= 0);
    (*s)--;
    return 0;
}

int Signal(int *s) {
    (*s)++;
    return 0;
}

void* philosopher(void* num) {
    int i = *(int*)num;

    do {
        printf("Philosopher %d is thinking\n", i);
        sleep(1);

        // Try to pick up left and right chopsticks
        wait(&mutex);
        wait(&chopstick[i]); // left chopstick
        wait(&chopstick[(i + 1) % N]); // right chopstick

        printf("Philosopher %d is eating\n", i);
        sleep(2); // eating

        // Put down chopsticks
        Signal(&chopstick[i]);
        Signal(&chopstick[(i + 1) % N]);
        Signal(&mutex);

        printf("Philosopher %d put down chopsticks and is thinking again\n", i);

        sleep(1);
    } while (1);

    return NULL;
}

int main() {
    pthread_t thread_id[N];
    int philosopher_ids[N];

    for (int i = 0; i < N; i++) {
        philosopher_ids[i] = i;
        pthread_create(&thread_id[i], NULL, philosopher, &philosopher_ids[i]);
    }

    for (int i = 0; i < N; i++) {
        pthread_join(thread_id[i], NULL);
    }

    return 0;
}
```



```
Philosopher 0 is thinking
Philosopher 1 is thinking
Philosopher 2 is thinking
Philosopher 3 is thinking
Philosopher 4 is thinking
Philosopher 1 is eating
Philosopher 1 put down chopsticks and is thinking again
Philosopher 4 is eating
Philosopher 2 is eating
Philosopher 1 is thinking
Philosopher 4 put down chopsticks and is thinking again
```