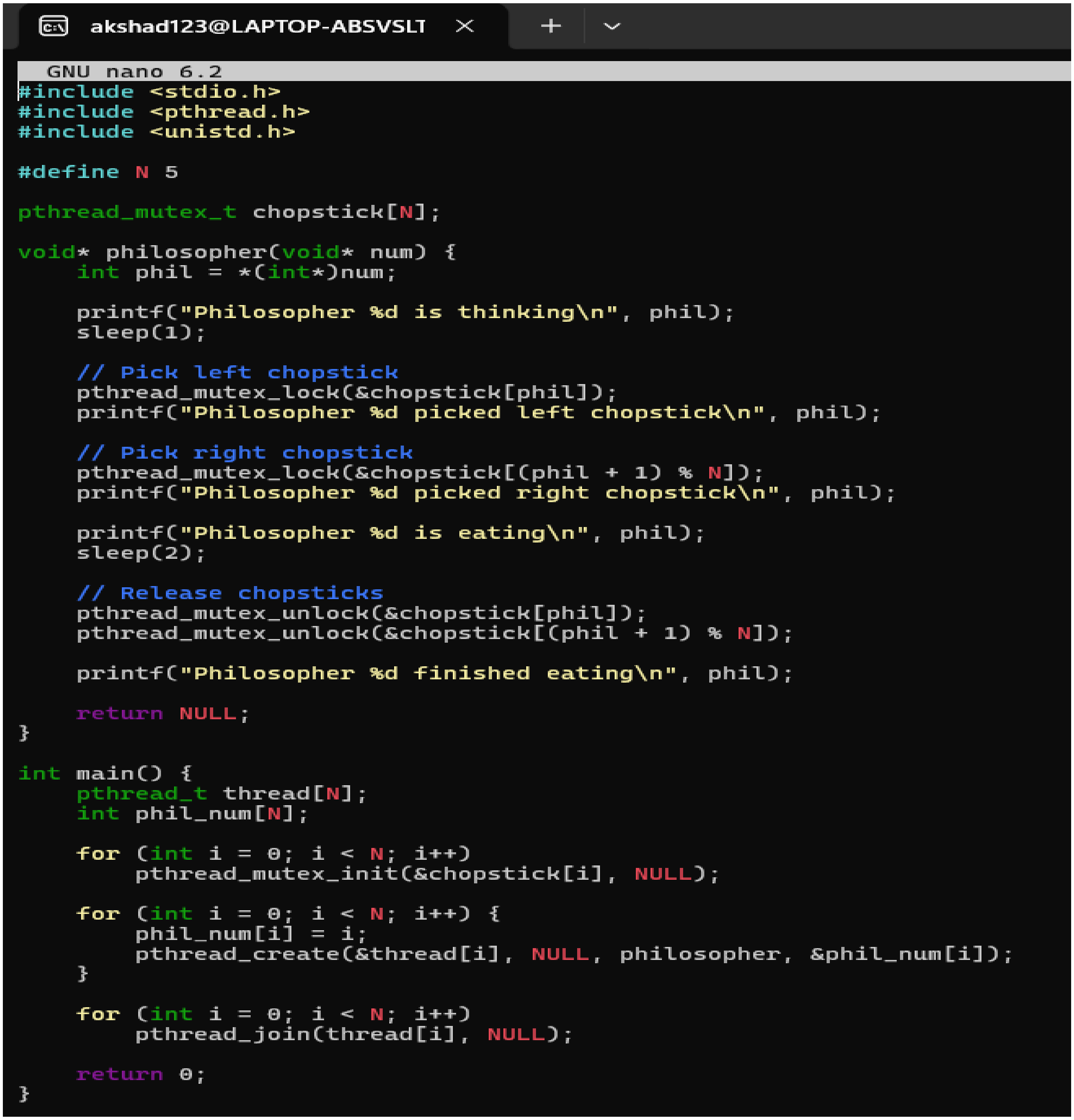
PRACTICAL NO. 6

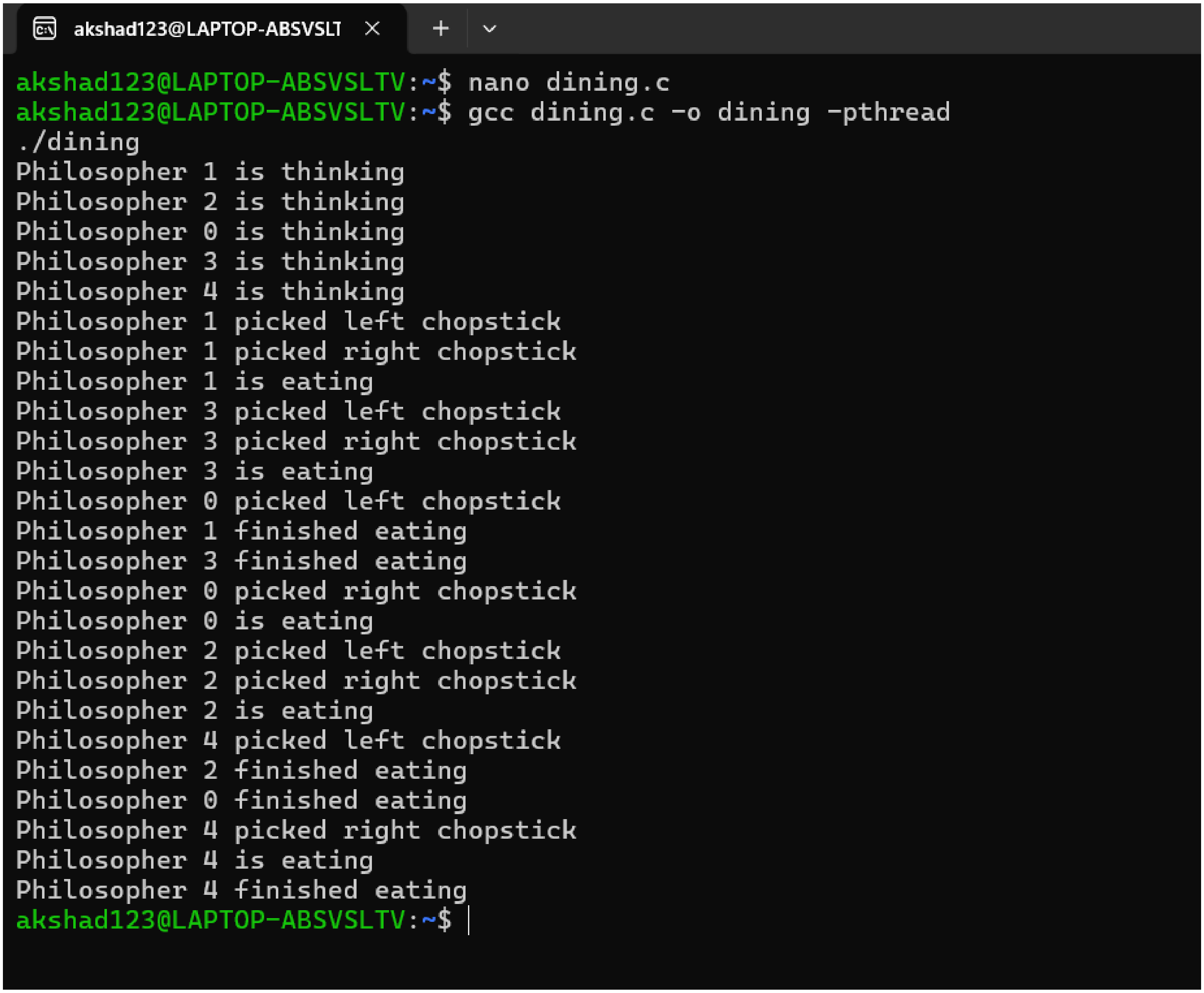
AIM: Considered there are N philosophers seated around a circular table with one chopstick between each pair of philosophers. There is one chopstick between each philosopher. A philosopher may eat if he can pick up the two chopsticks adjacent to him. one chopstick may be picked up by any one of its adjacent followers but not both. Write a program to solve the problem using process synchronization technique.

CODE :



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

#define N 5

pthread_mutex_t chopstick[N];

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

    printf("Philosopher %d is thinking\n", phil);
    sleep(1);

    // Pick left chopstick
    pthread_mutex_lock(&chopstick[phil]);
    printf("Philosopher %d picked left chopstick\n", phil);

    // Pick right chopstick
    pthread_mutex_lock(&chopstick[(phil + 1) % N]);
    printf("Philosopher %d picked right chopstick\n", phil);

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

    // Release chopsticks
    pthread_mutex_unlock(&chopstick[phil]);
    pthread_mutex_unlock(&chopstick[(phil + 1) % N]);

    printf("Philosopher %d finished eating\n", phil);

    return NULL;
}

int main() {
    pthread_t thread[N];
    int phil_num[N];

    for (int i = 0; i < N; i++)
        pthread_mutex_init(&chopstick[i], NULL);

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

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

    return 0;
}
```

OUTPUT:



```
akshad123@LAPTOP-ABSVSLTV:~$ nano dining.c
akshad123@LAPTOP-ABSVSLTV:~$ gcc dining.c -o dining -pthread
./dining
Philosopher 1 is thinking
Philosopher 2 is thinking
Philosopher 0 is thinking
Philosopher 3 is thinking
Philosopher 4 is thinking
Philosopher 1 picked left chopstick
Philosopher 1 picked right chopstick
Philosopher 1 is eating
Philosopher 3 picked left chopstick
Philosopher 3 picked right chopstick
Philosopher 3 is eating
Philosopher 0 picked left chopstick
Philosopher 1 finished eating
Philosopher 3 finished eating
Philosopher 0 picked right chopstick
Philosopher 0 is eating
Philosopher 2 picked left chopstick
Philosopher 2 picked right chopstick
Philosopher 2 is eating
Philosopher 4 picked left chopstick
Philosopher 2 finished eating
Philosopher 0 finished eating
Philosopher 4 picked right chopstick
Philosopher 4 is eating
Philosopher 4 finished eating
akshad123@LAPTOP-ABSVSLTV:~$
```