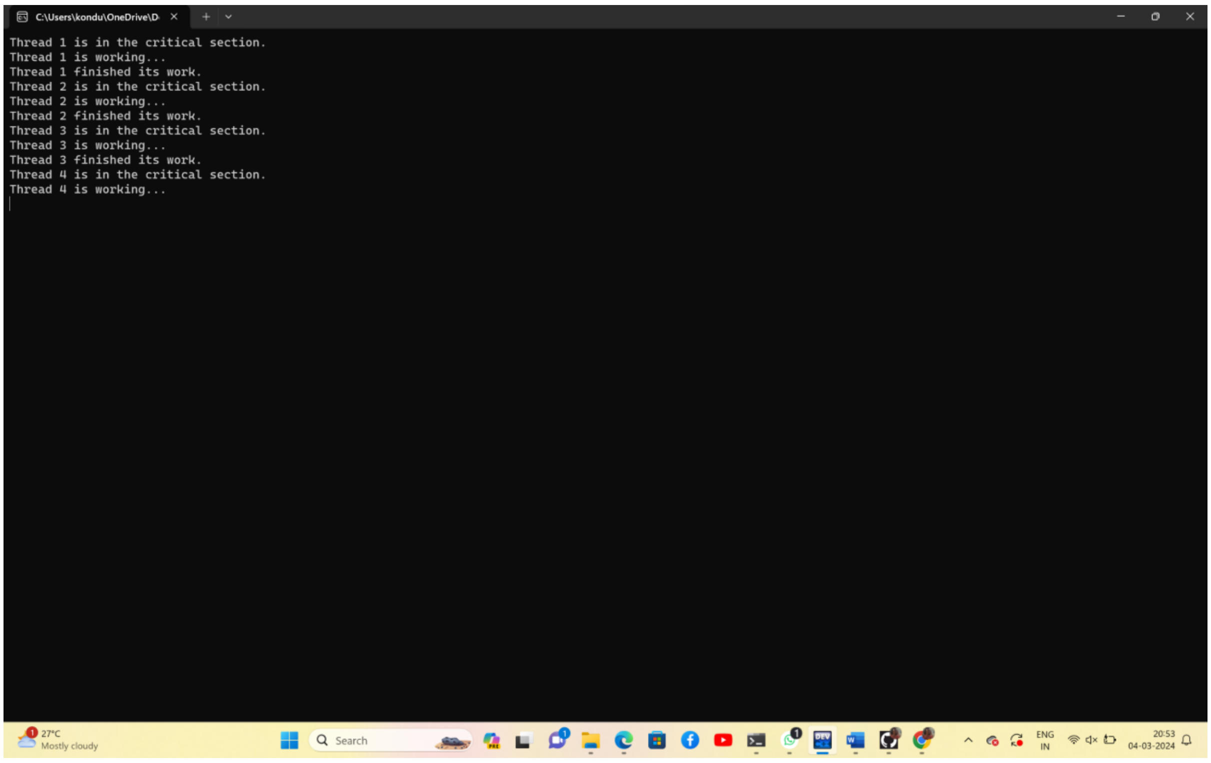
```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
#include <unistd.h> // Include this header for the sleep function

#define NUM_THREADS 5

pthread_mutex_t mutex_lock; // Define a mutex lock

void *thread_function(void *arg) {
    int thread_id = *((int *)arg);

    // Lock the critical section
    pthread_mutex_lock(&mutex_lock);

    printf("Thread %d is in the critical section.\n", thread_id);

    // Simulate some work being done
    printf("Thread %d is working...\n", thread_id);
    sleep(2);

    printf("Thread %d finished its work.\n", thread_id);

    // Unlock the critical section
    pthread_mutex_unlock(&mutex_lock);

    pthread_exit(NULL);
}

int main() {
    pthread_t threads[NUM_THREADS];
    int thread_args[NUM_THREADS];
    int i;

    // Initialize the mutex lock
    pthread_mutex_init(&mutex_lock, NULL);

    // Create threads
    for (i = 0; i < NUM_THREADS; i++) {
        thread_args[i] = i + 1;
        if (pthread_create(&threads[i], NULL, thread_function, &thread_args[i]) != 0) {
            fprintf(stderr, "Error creating thread %d\n", i);
            exit(EXIT_FAILURE);
        }
    }

    // Join threads
    for (i = 0; i < NUM_THREADS; i++) {
        if (pthread_join(threads[i], NULL) != 0) {
            fprintf(stderr, "Error joining thread %d\n", i);
            exit(EXIT_FAILURE);
        }
    }

    // Destroy the mutex lock
    pthread_mutex_destroy(&mutex_lock);

    return 0;
}
```



```
Thread 1 is in the critical section.
Thread 1 is working...
Thread 1 finished its work.
Thread 2 is in the critical section.
Thread 2 is working...
Thread 2 finished its work.
Thread 3 is in the critical section.
Thread 3 is working...
Thread 3 finished its work.
Thread 4 is in the critical section.
Thread 4 is working...
```