National Textile University

Department of Computer Science

Subject:

Operating System

Submitted To:

Sir Nasir Mehmood

Submitted By:

Hafsa Tayyab

Registration No:

23-NTU-CS-1163

Lab No:

5

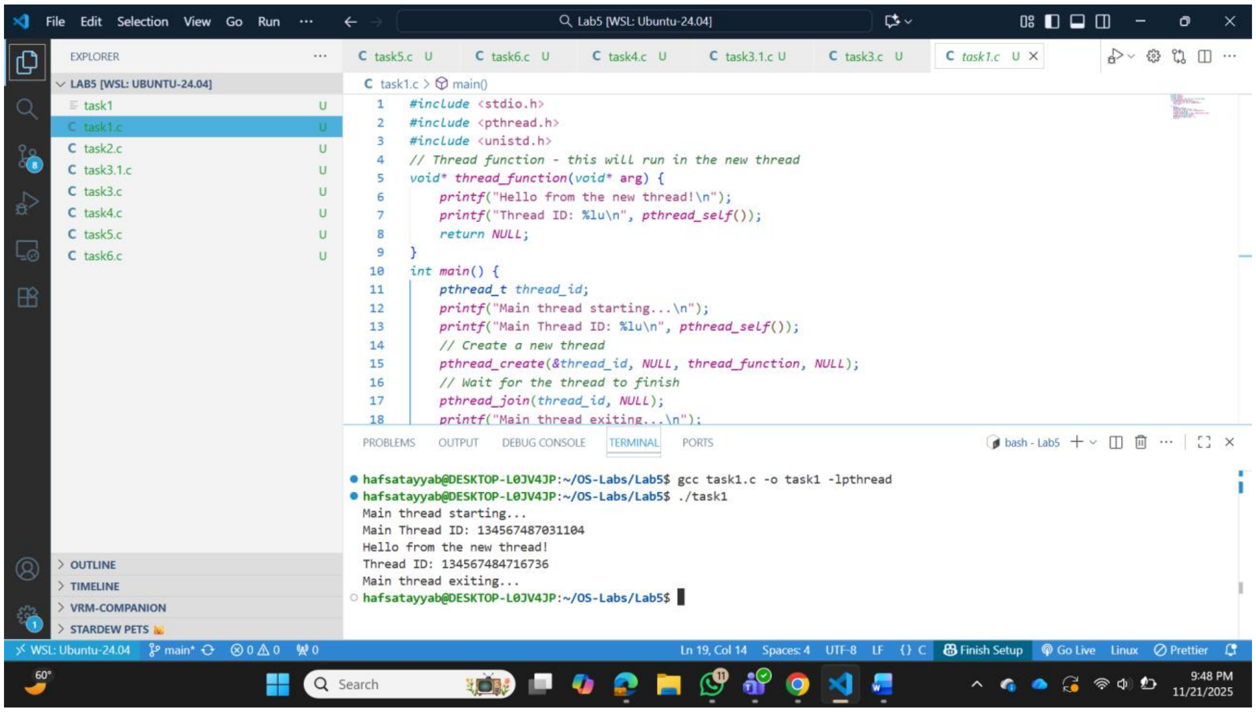
Semester:

5th

Lab 5: Introduction to Threads

3. C Programs with Threads

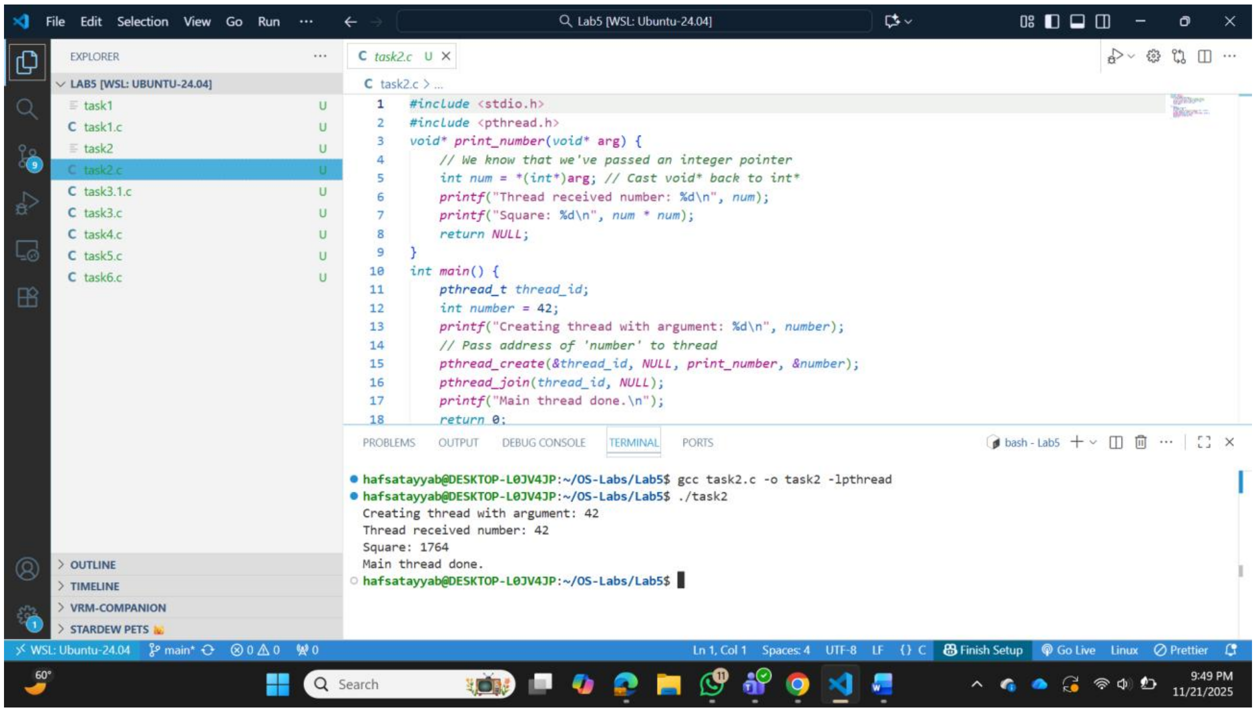
Program 1: Creating a Simple Thread



```c
#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
// Thread function - this will run in the new thread
void* thread_function(void* arg) {
    printf("Hello from the new thread!\n");
    printf("Thread ID: %lu\n", pthread_self());
    return NULL;
}
int main() {
    pthread_t thread_id;
    printf("Main thread starting...\n");
    printf("Main Thread ID: %lu\n", pthread_self());
    // Create a new thread
    pthread_create(&thread_id, NULL, thread_function, NULL);
    // Wait for the thread to finish
    pthread_join(thread_id, NULL);
    printf("Main thread exiting...\n");
```

```
hafsatayyab@DESKTOP-L0JV4JP:~/OS-Labs/Lab5$ gcc task1.c -o task1 -lpthread
hafsatayyab@DESKTOP-L0JV4JP:~/OS-Labs/Lab5$ ./task1
Main thread starting...
Main Thread ID: 134567487031104
Hello from the new thread!
Thread ID: 134567484716736
Main thread exiting...
hafsatayyab@DESKTOP-L0JV4JP:~/OS-Labs/Lab5$
```

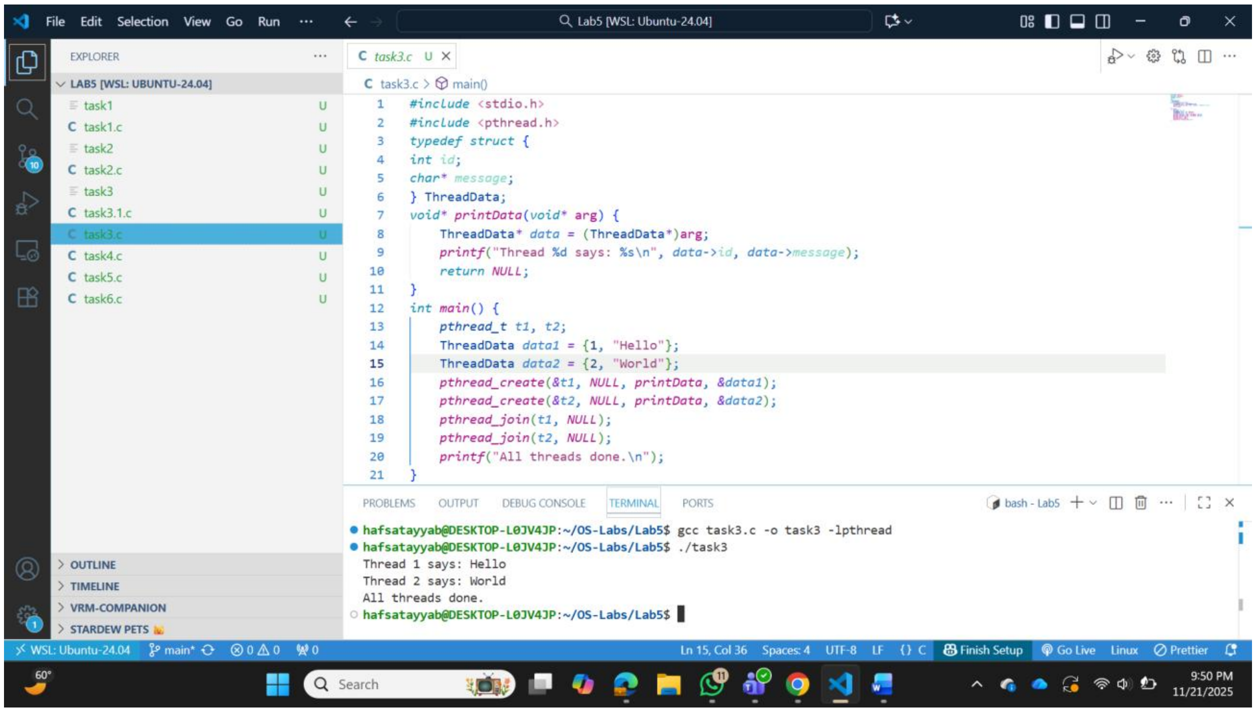
Program 2: Passing Arguments to Threads



```c
#include <stdio.h>
#include <pthread.h>
void* print_number(void* arg) {
    // We know that we've passed an integer pointer
    int num = *(int*)arg; // Cast void* back to int*
    printf("Thread received number: %d\n", num);
    printf("Square: %d\n", num * num);
    return NULL;
}
int main() {
    pthread_t thread_id;
    int number = 42;
    printf("Creating thread with argument: %d\n", number);
    // Pass address of 'number' to thread
    pthread_create(&thread_id, NULL, print_number, &number);
    pthread_join(thread_id, NULL);
    printf("Main thread done.\n");
    return 0;
```

```
hafsatayyab@DESKTOP-L0JV4JP:~/OS-Labs/Lab5$ gcc task2.c -o task2 -lpthread
hafsatayyab@DESKTOP-L0JV4JP:~/OS-Labs/Lab5$ ./task2
Creating thread with argument: 42
Thread received number: 42
Square: 1764
Main thread done.
hafsatayyab@DESKTOP-L0JV4JP:~/OS-Labs/Lab5$
```

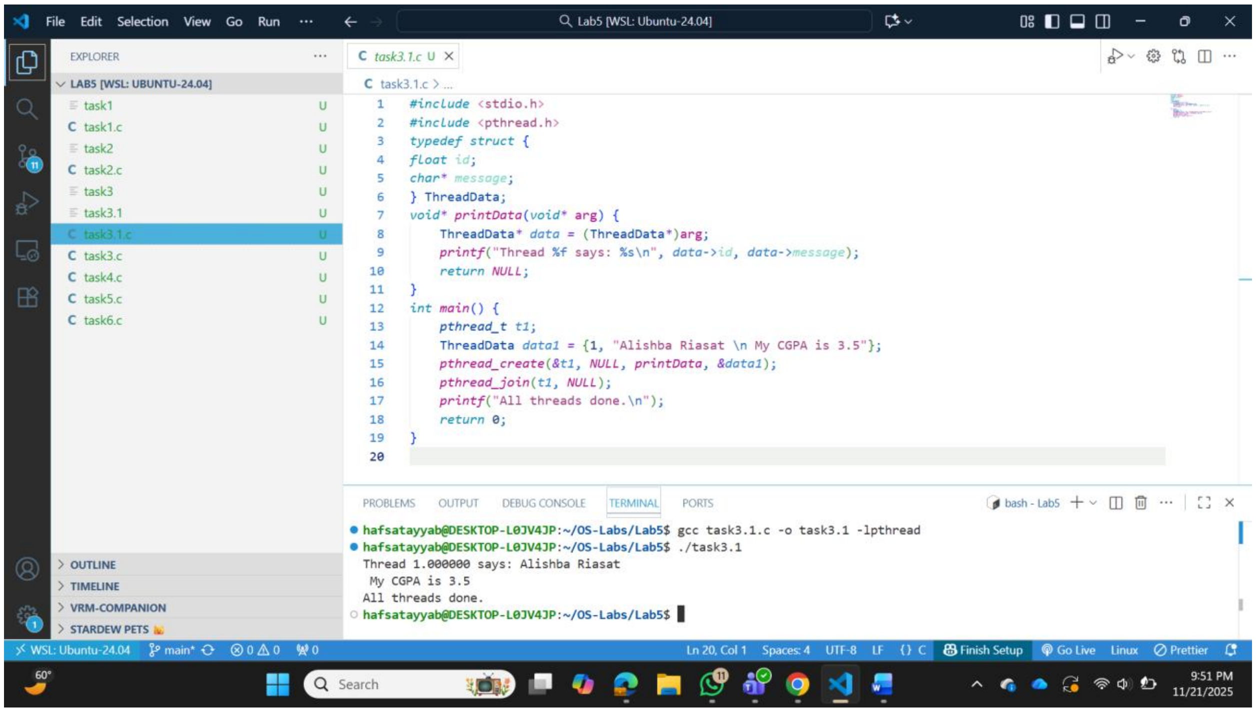
Program 3: Passing Multiple Data



```c
#include <stdio.h>
#include <pthread.h>
typedef struct {
int id;
char* message;
} ThreadData;
void* printData(void* arg) {
    ThreadData* data = (ThreadData*)arg;
    printf("Thread %d says: %s\n", data->id, data->message);
    return NULL;
}
int main() {
    pthread_t t1, t2;
    ThreadData data1 = {1, "Hello"};
    ThreadData data2 = {2, "World"};
    pthread_create(&t1, NULL, printData, &data1);
    pthread_create(&t2, NULL, printData, &data2);
    pthread_join(t1, NULL);
    pthread_join(t2, NULL);
    printf("All threads done.\n");
}
```

```
hafsatayyab@DESKTOP-L0JV4JP:~/OS-Labs/Lab5$ gcc task3.c -o task3 -lpthread
hafsatayyab@DESKTOP-L0JV4JP:~/OS-Labs/Lab5$ ./task3
Thread 1 says: Hello
Thread 2 says: World
All threads done.
hafsatayyab@DESKTOP-L0JV4JP:~/OS-Labs/Lab5$
```

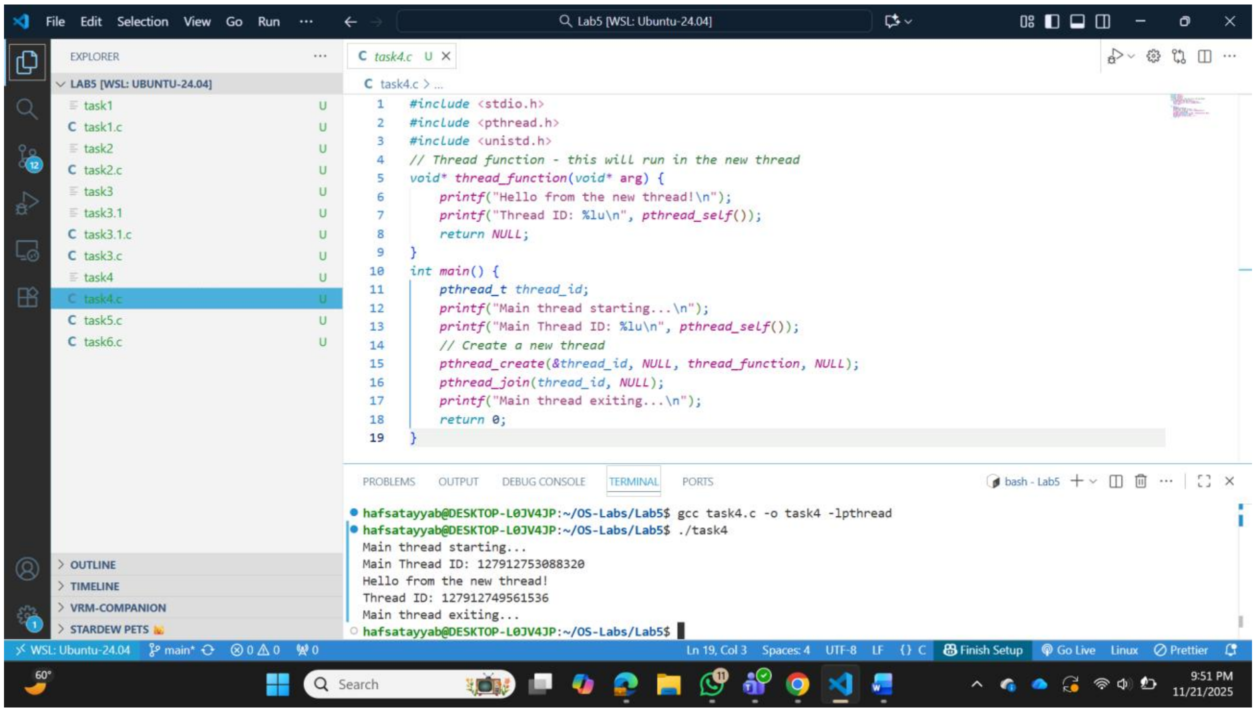
Task 3.1 (Name & Cgpa)



```c
#include <stdio.h>
#include <pthread.h>
typedef struct {
float id;
char* message;
} ThreadData;
void* printData(void* arg) {
    ThreadData* data = (ThreadData*)arg;
    printf("Thread %f says: %s\n", data->id, data->message);
    return NULL;
}
int main() {
    pthread_t t1;
    ThreadData data1 = {1, "Alishba Riasat \n My CGPA is 3.5"};
    pthread_create(&t1, NULL, printData, &data1);
    pthread_join(t1, NULL);
    printf("All threads done.\n");
    return 0;
}
```

```
hafsatayyab@DESKTOP-L0JV4JP:~/OS-Labs/Lab5$ gcc task3.1.c -o task3.1 -lpthread
hafsatayyab@DESKTOP-L0JV4JP:~/OS-Labs/Lab5$ ./task3.1
Thread 1.000000 says: Alishba Riasat
 My CGPA is 3.5
All threads done.
hafsatayyab@DESKTOP-L0JV4JP:~/OS-Labs/Lab5$
```

Program 4: Thread Return Values

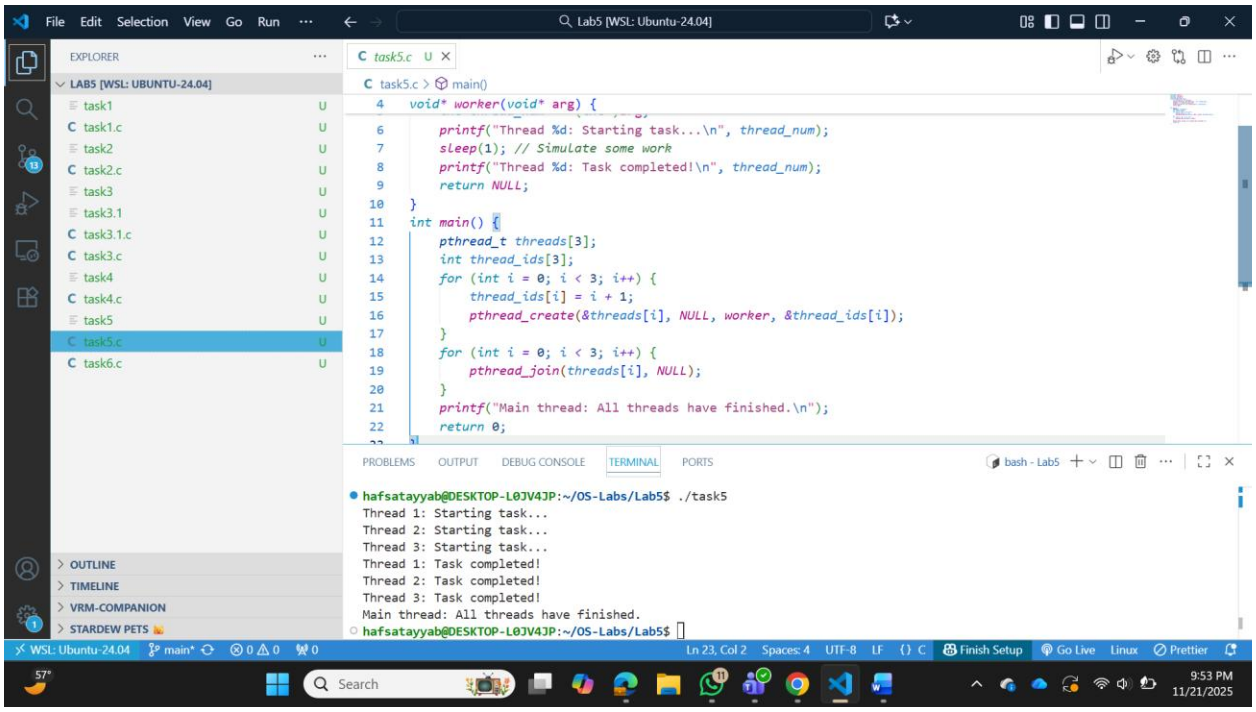


```c
#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
// Thread function - this will run in the new thread
void* thread_function(void* arg) {
    printf("Hello from the new thread!\n");
    printf("Thread ID: %lu\n", pthread_self());
    return NULL;
}
int main() {
    pthread_t thread_id;
    printf("Main thread starting...\n");
    printf("Main Thread ID: %lu\n", pthread_self());
    // Create a new thread
    pthread_create(&thread_id, NULL, thread_function, NULL);
    pthread_join(thread_id, NULL);
    printf("Main thread exiting...\n");
    return 0;
}
```

```
hafsatayyab@DESKTOP-L0JV4JP:~/OS-Labs/Lab5$ gcc task4.c -o task4 -lpthread
hafsatayyab@DESKTOP-L0JV4JP:~/OS-Labs/Lab5$ ./task4
Main thread starting...
Main Thread ID: 127912753088320
Hello from the new thread!
Thread ID: 127912749561536
Main thread exiting...
hafsatayyab@DESKTOP-L0JV4JP:~/OS-Labs/Lab5$
```

4. Basic Multithreading

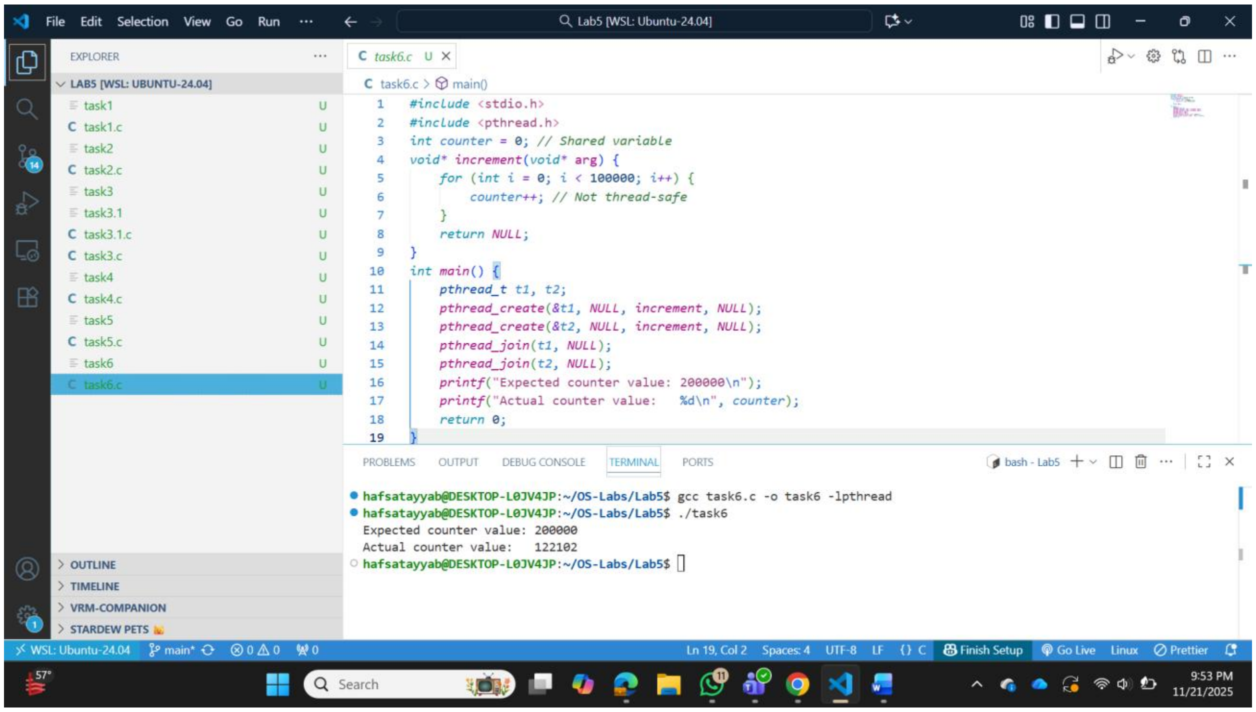
Program 1: Creating and Running Multiple Threads



```c
void* worker(void* arg) {
    printf("Thread %d: Starting task...\n", thread_num);
    sleep(1); // Simulate some work
    printf("Thread %d: Task completed!\n", thread_num);
    return NULL;
}
int main() {
    pthread_t threads[3];
    int thread_ids[3];
    for (int i = 0; i < 3; i++) {
        thread_ids[i] = i + 1;
        pthread_create(&threads[i], NULL, worker, &thread_ids[i]);
    }
    for (int i = 0; i < 3; i++) {
        pthread_join(threads[i], NULL);
    }
    printf("Main thread: All threads have finished.\n");
    return 0;
```

```
hafsatayyab@DESKTOP-L0JV4JP:~/OS-Labs/Lab5$ ./task5
Thread 1: Starting task...
Thread 2: Starting task...
Thread 3: Starting task...
Thread 1: Task completed!
Thread 2: Task completed!
Thread 3: Task completed!
Main thread: All threads have finished.
hafsatayyab@DESKTOP-L0JV4JP:~/OS-Labs/Lab5$
```

Program 2: Demonstrating a Race Condition



```c
#include <stdio.h>
#include <pthread.h>
int counter = 0; // Shared variable
void* increment(void* arg) {
    for (int i = 0; i < 100000; i++) {
        counter++; // Not thread-safe
    }
    return NULL;
}
int main() {
    pthread_t t1, t2;
    pthread_create(&t1, NULL, increment, NULL);
    pthread_create(&t2, NULL, increment, NULL);
    pthread_join(t1, NULL);
    pthread_join(t2, NULL);
    printf("Expected counter value: 200000\n");
    printf("Actual counter value:   %d\n", counter);
    return 0;
}
```

```
hafsatayyab@DESKTOP-L0JV4JP:~/OS-Labs/Lab5$ gcc task6.c -o task6 -lpthread
hafsatayyab@DESKTOP-L0JV4JP:~/OS-Labs/Lab5$ ./task6
Expected counter value: 200000
Actual counter value:   122102
hafsatayyab@DESKTOP-L0JV4JP:~/OS-Labs/Lab5$
```