



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

```
1 #include <stdio.h>
2 #include <pthread.h>
3 #include <unistd.h>
4 // Thread function - this will run in the new thread
5 void* thread_function(void* arg) {
6     printf("Hello from the new thread!\n");
7     printf("Thread ID: %lu\n", pthread_self());
8     return NULL;
9 }
10 int main() {
11     pthread_t thread_id;
12     printf("Main thread starting...\n");
13     printf("Main Thread ID: %lu\n", pthread_self());
14     // Create a new thread
15     pthread_create(&thread_id, NULL, thread_function, NULL);
16     // Wait for the thread to finish
17     pthread_join(thread_id, NULL);
18     printf("Main thread exiting...\n");
19 }
```

```
hafsatayyab@DESKTOP-L0JV43P:~/OS-Labs/Lab5$ gcc task1.c -o task1 -lpthread
hafsatayyab@DESKTOP-L0JV43P:~/OS-Labs/Lab5$ ./task1
Main thread starting...
Main Thread ID: 134567487831104
Hello from the new thread!
Thread ID: 134567484716736
Main thread exiting...
```

Program 2: Passing Arguments to Threads

```
1 #include <stdio.h>
2 #include <pthread.h>
3 void* print_number(void* arg) {
4     // We know that we've passed an integer pointer
5     int num = *(int*)arg; // Cast void* back to int*
6     printf("Thread received number: %d\n", num);
7     printf("Square: %d\n", num * num);
8     return NULL;
9 }
10 int main() {
11     pthread_t thread_id;
12     int number = 42;
13     printf("Creating thread with argument: %d\n", number);
14     // Pass address of 'number' to thread
15     pthread_create(&thread_id, NULL, print_number, &number);
16     pthread_join(thread_id, NULL);
17     printf("Main thread done.\n");
18     return 0;
19 }
```

```
hafsatayyab@DESKTOP-L0JV43P:~/OS-Labs/Lab5$ gcc task2.c -o task2 -lpthread
hafsatayyab@DESKTOP-L0JV43P:~/OS-Labs/Lab5$ ./task2
Creating thread with argument: 42
Thread received number: 42
Square: 1764
Main thread done.
```

Program 3: Passing Multiple Data

The screenshot shows the Visual Studio Code editor with the file explorer on the left displaying a project named 'LAB5 [WSL: UBUNTU-24.04]' containing files 'task1.c' through 'task6.c'. The main editor window shows 'task3.c' with the following C code:

```
1 #include <stdio.h>
2 #include <pthread.h>
3 typedef struct {
4     int id;
5     char* message;
6 } ThreadData;
7 void* printData(void* arg) {
8     ThreadData* data = (ThreadData*)arg;
9     printf("Thread %d says: %s\n", data->id, data->message);
10    return NULL;
11 }
12 int main() {
13     pthread_t t1, t2;
14     ThreadData data1 = {1, "Hello"};
15     ThreadData data2 = {2, "World"};
16     pthread_create(&t1, NULL, printData, &data1);
17     pthread_create(&t2, NULL, printData, &data2);
18     pthread_join(t1, NULL);
19     pthread_join(t2, NULL);
20     printf("All threads done.\n");
21 }
```

The terminal at the bottom shows the execution of the program:

```
hafsatayyab@DESKTOP-L0JV43P:~/OS-Labs/Lab5$ gcc task3.c -o task3 -lpthread
hafsatayyab@DESKTOP-L0JV43P:~/OS-Labs/Lab5$ ./task3
Thread 1 says: Hello
Thread 2 says: World
All threads done.
hafsatayyab@DESKTOP-L0JV43P:~/OS-Labs/Lab5$
```

Task 3.1 (Name & Cgpa)

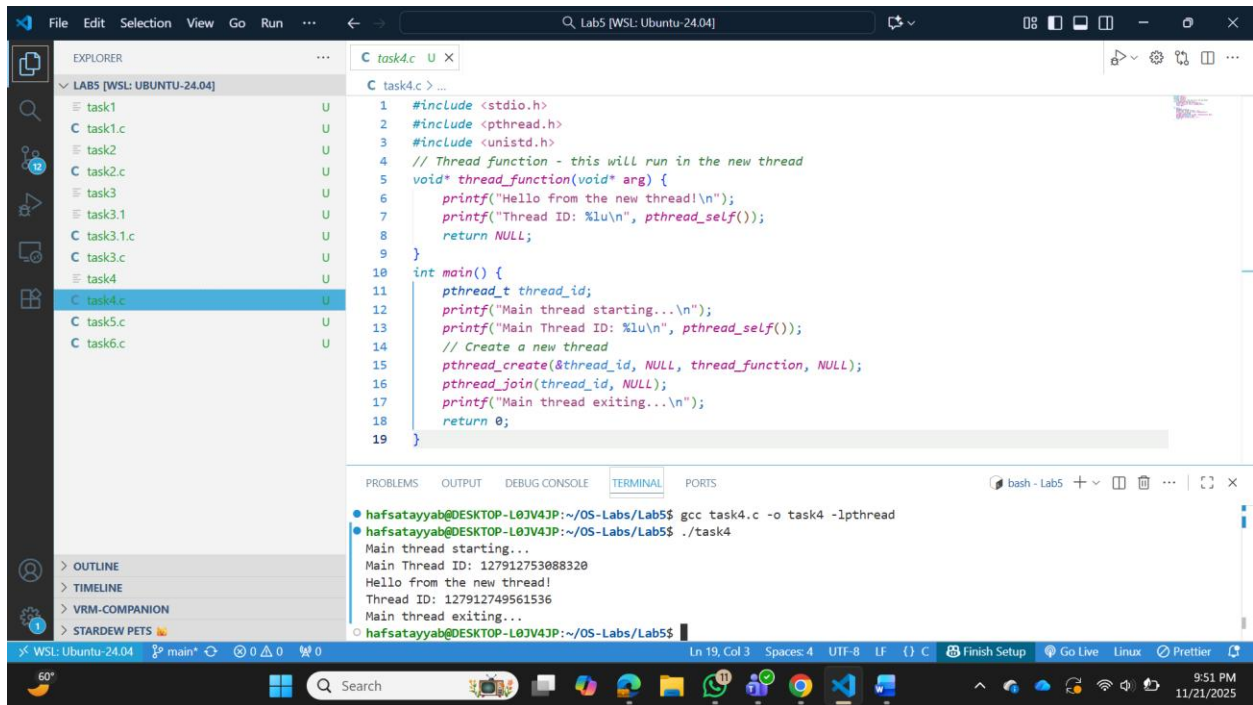
The screenshot shows the Visual Studio Code editor with the file explorer on the left displaying the same project 'LAB5 [WSL: UBUNTU-24.04]' with files 'task1.c' through 'task6.c'. The main editor window shows 'task3.1.c' with the following C code:

```
1 #include <stdio.h>
2 #include <pthread.h>
3 typedef struct {
4     float id;
5     char* message;
6 } ThreadData;
7 void* printData(void* arg) {
8     ThreadData* data = (ThreadData*)arg;
9     printf("Thread %f says: %s\n", data->id, data->message);
10    return NULL;
11 }
12 int main() {
13     pthread_t t1;
14     ThreadData data1 = {1, "Alishba Riasat \n My CGPA is 3.5"};
15     pthread_create(&t1, NULL, printData, &data1);
16     pthread_join(t1, NULL);
17     printf("All threads done.\n");
18     return 0;
19 }
20
```

The terminal at the bottom shows the execution of the program:

```
hafsatayyab@DESKTOP-L0JV43P:~/OS-Labs/Lab5$ gcc task3.1.c -o task3.1 -lpthread
hafsatayyab@DESKTOP-L0JV43P:~/OS-Labs/Lab5$ ./task3.1
Thread 1.000000 says: Alishba Riasat
My CGPA is 3.5
All threads done.
hafsatayyab@DESKTOP-L0JV43P:~/OS-Labs/Lab5$
```

Program 4: Thread Return Values



The screenshot displays the Visual Studio Code editor with a C program named `task4.c` open. The program is designed to demonstrate thread creation and return values. It includes standard headers (`<stdio.h>`, `<pthread.h>`, `<unistd.h>`) and defines a `thread_function` that prints a message and its thread ID before returning `NULL`. The `main` function creates a new thread using `pthread_create`, joins it with `pthread_join`, and prints the main thread's ID and exit message.

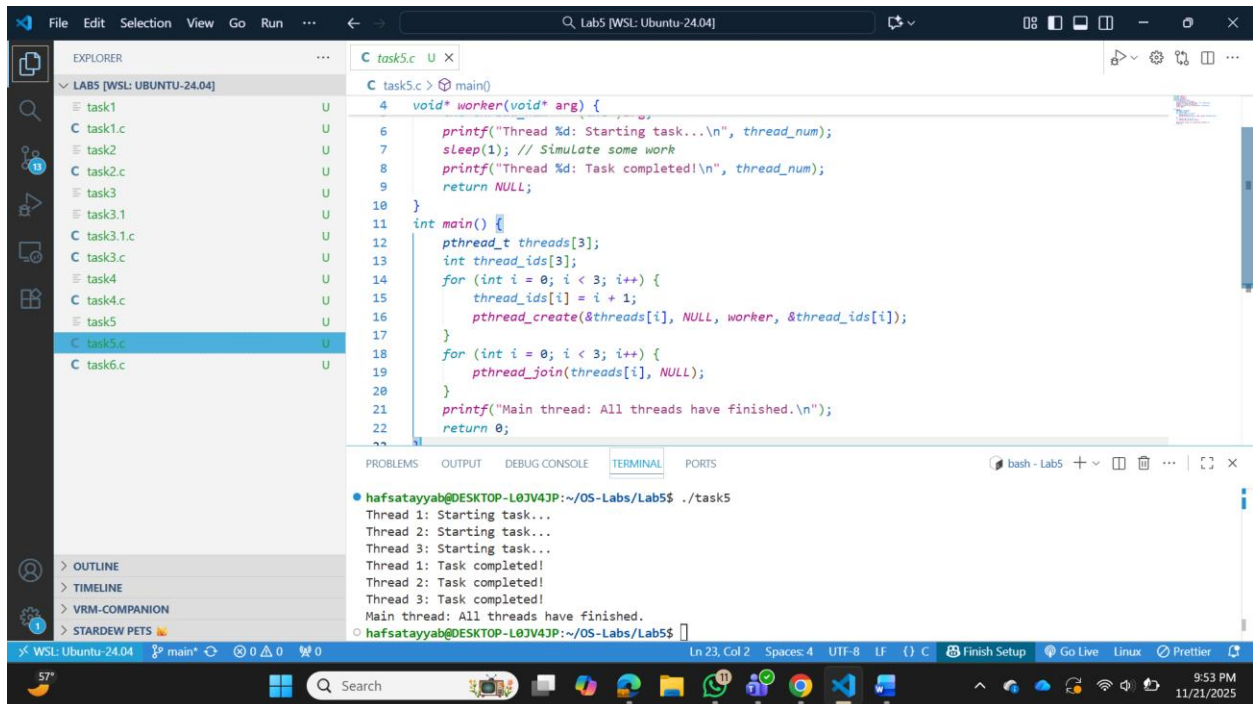
```
1 #include <stdio.h>
2 #include <pthread.h>
3 #include <unistd.h>
4 // Thread function - this will run in the new thread
5 void* thread_function(void* arg) {
6     printf("Hello from the new thread!\n");
7     printf("Thread ID: %lu\n", pthread_self());
8     return NULL;
9 }
10 int main() {
11     pthread_t thread_id;
12     printf("Main thread starting...\n");
13     printf("Main Thread ID: %lu\n", pthread_self());
14     // Create a new thread
15     pthread_create(&thread_id, NULL, thread_function, NULL);
16     pthread_join(thread_id, NULL);
17     printf("Main thread exiting...\n");
18     return 0;
19 }
```

The terminal output shows the successful execution of the program. It first runs `gcc task4.c -o task4 -lpthread` to compile the code, then executes `./task4`. The output matches the program's logic, showing the main thread starting, the new thread printing its message and ID, and the main thread exiting.

```
hafsatayyab@DESKTOP-L0JV43P:~/OS-Labs/Lab5$ gcc task4.c -o task4 -lpthread
hafsatayyab@DESKTOP-L0JV43P:~/OS-Labs/Lab5$ ./task4
Main thread starting...
Main Thread ID: 127912753088320
Hello from the new thread!
Thread ID: 127912749561536
Main thread exiting...
hafsatayyab@DESKTOP-L0JV43P:~/OS-Labs/Lab5$
```

4. Basic Multithreading

Program 1: Creating and Running Multiple Threads

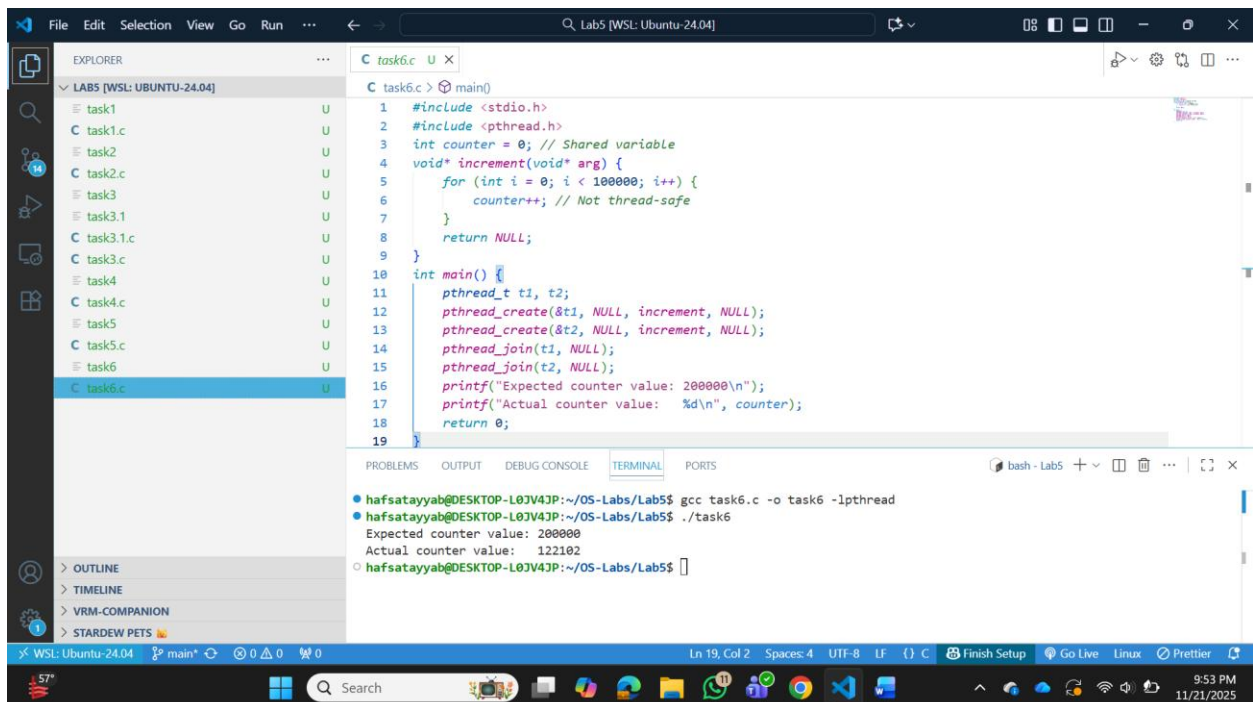


```
File Edit Selection View Go Run ... C task5.c U X
LABS [WSL: UBUNTU-24.04]
task1 U
task1.c U
task2 U
task2.c U
task3 U
task3.1 U
task3.1.c U
task3.c U
task4 U
task4.c U
task5 U
task5.c U
task6.c U

C task5.c > main()
4 void* worker(void* arg) {
6     printf("Thread %d: Starting task...\n", thread_num);
7     sleep(1); // Simulate some work
8     printf("Thread %d: Task completed!\n", thread_num);
9     return NULL;
10 }
11 int main() {
12     pthread_t threads[3];
13     int thread_ids[3];
14     for (int i = 0; i < 3; i++) {
15         thread_ids[i] = i + 1;
16         pthread_create(&threads[i], NULL, worker, &thread_ids[i]);
17     }
18     for (int i = 0; i < 3; i++) {
19         pthread_join(threads[i], NULL);
20     }
21     printf("Main thread: All threads have finished.\n");
22     return 0;
23 }

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
hafsatayyab@DESKTOP-L0JV43P:~/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-L0JV43P:~/OS-Labs/Lab5$
```

Program 2: Demonstrating a Race Condition



```
File Edit Selection View Go Run ... C task6.c U X
LABS [WSL: UBUNTU-24.04]
task1 U
task1.c U
task2 U
task2.c U
task3 U
task3.1 U
task3.1.c U
task3.c U
task4 U
task4.c U
task5 U
task5.c U
task6 U
task6.c U

C task6.c > main()
1 #include <stdio.h>
2 #include <pthread.h>
3 int counter = 0; // Shared variable
4 void* increment(void* arg) {
5     for (int i = 0; i < 100000; i++) {
6         counter++; // Not thread-safe
7     }
8     return NULL;
9 }
10 int main() {
11     pthread_t t1, t2;
12     pthread_create(&t1, NULL, increment, NULL);
13     pthread_create(&t2, NULL, increment, NULL);
14     pthread_join(t1, NULL);
15     pthread_join(t2, NULL);
16     printf("Expected counter value: 200000\n");
17     printf("Actual counter value: %d\n", counter);
18     return 0;
19 }

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
hafsatayyab@DESKTOP-L0JV43P:~/OS-Labs/Lab5$ gcc task6.c -o task6 -lpthread
hafsatayyab@DESKTOP-L0JV43P:~/OS-Labs/Lab5$ ./task6
Expected counter value: 200000
Actual counter value: 122102
hafsatayyab@DESKTOP-L0JV43P:~/OS-Labs/Lab5$
```