National Textile University

Department of Computer Science

Subject: Operating System

Submitted to: Sir Nasir

Submitted by: Maha

Reg. number: 23-NTU-CS-1170

Lab 5(class task)

Semester:5th

3. C Programs with Threads

Program 1: Creating a Simple Thread

Objective: Create a thread and print messages from both main thread and new thread.

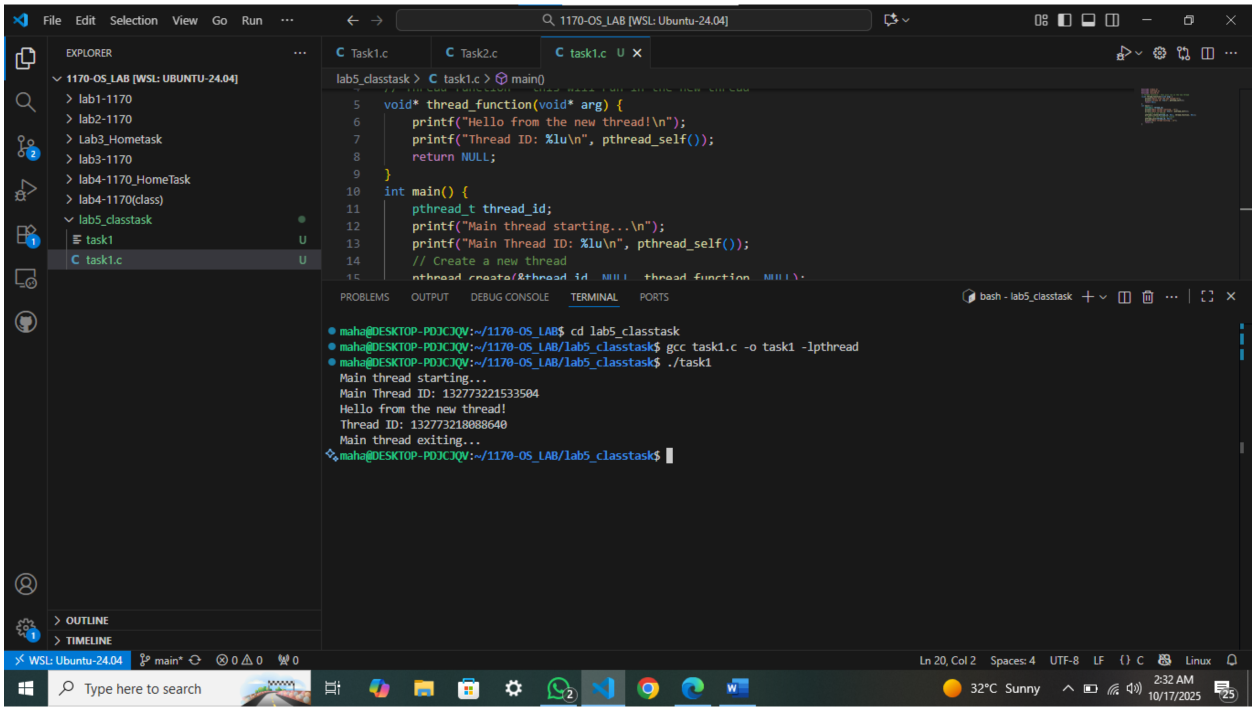
Code:

```
#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");

    return 0;

}
```

Execution commands:

```
gcc thread1.c -o thread1 -lpthread
./thread1
```

Terminal:



```c
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
```

```
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB$ cd lab5_classtask
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ gcc task1.c -o task1 -lpthread
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ ./task1
Main thread starting...
Main Thread ID: 132773221533504
Hello from the new thread!
Thread ID: 132773218088640
Main thread exiting...
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$
```

Program 2: Passing Arguments to Threads

Objective: Pass data to a thread function.

Code:

```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;

}
```

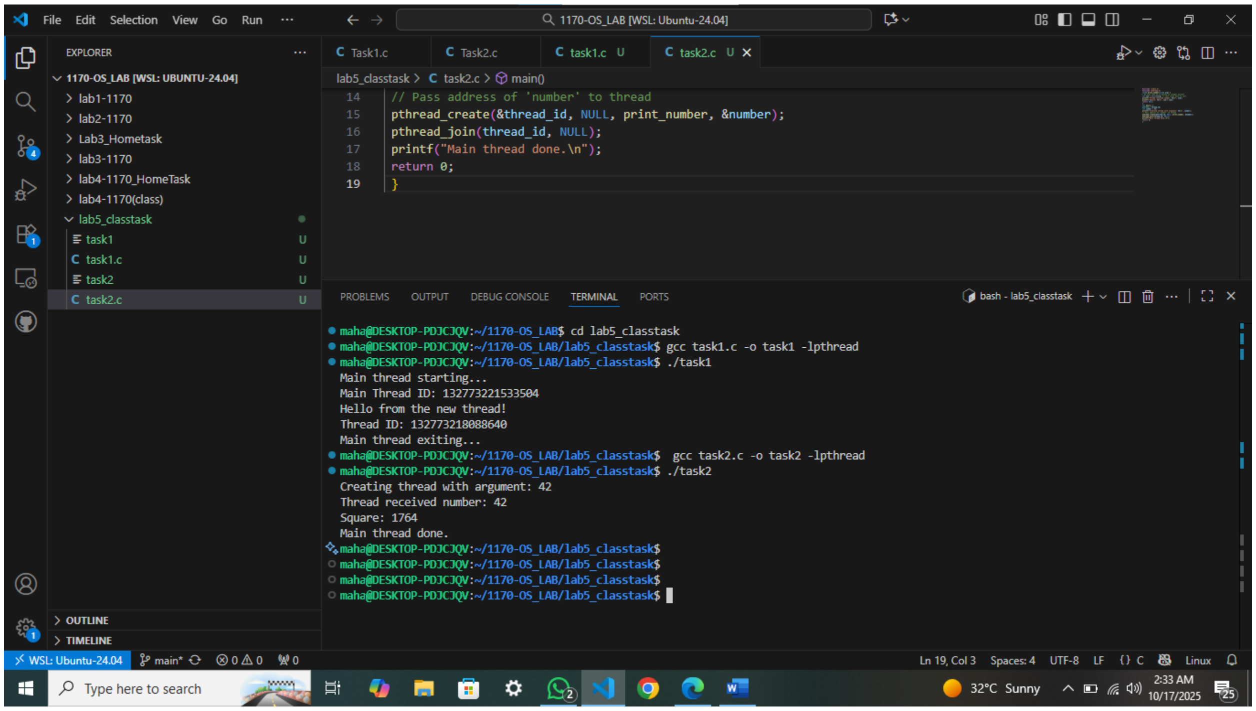
```c
int main() {
pthread_t thread_id;
int number = 42;
printf("Creating thread with argument: %d\n", number);
// Pass address of 'number' to thread
pthread_create(&thread_id, NULL, print_number, &number);
pthread_join(thread_id, NULL);
printf("Main thread done.\n");
return 0;
}
```

Execution commands:

Compile and run:

```
gcc thread2.c -o thread2 -lpthread
./thread2
```

Terminal:



```c
// 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;
}
```

```
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB$ cd lab5_classtask
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ gcc task1.c -o task1 -lpthread
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ ./task1
Main thread starting...
Main Thread ID: 132773221533504
Hello from the new thread!
Thread ID: 132773218088640
Main thread exiting...
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$  gcc task2.c -o task2 -lpthread
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ ./task2
Creating thread with argument: 42
Thread received number: 42
Square: 1764
Main thread done.
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$
```

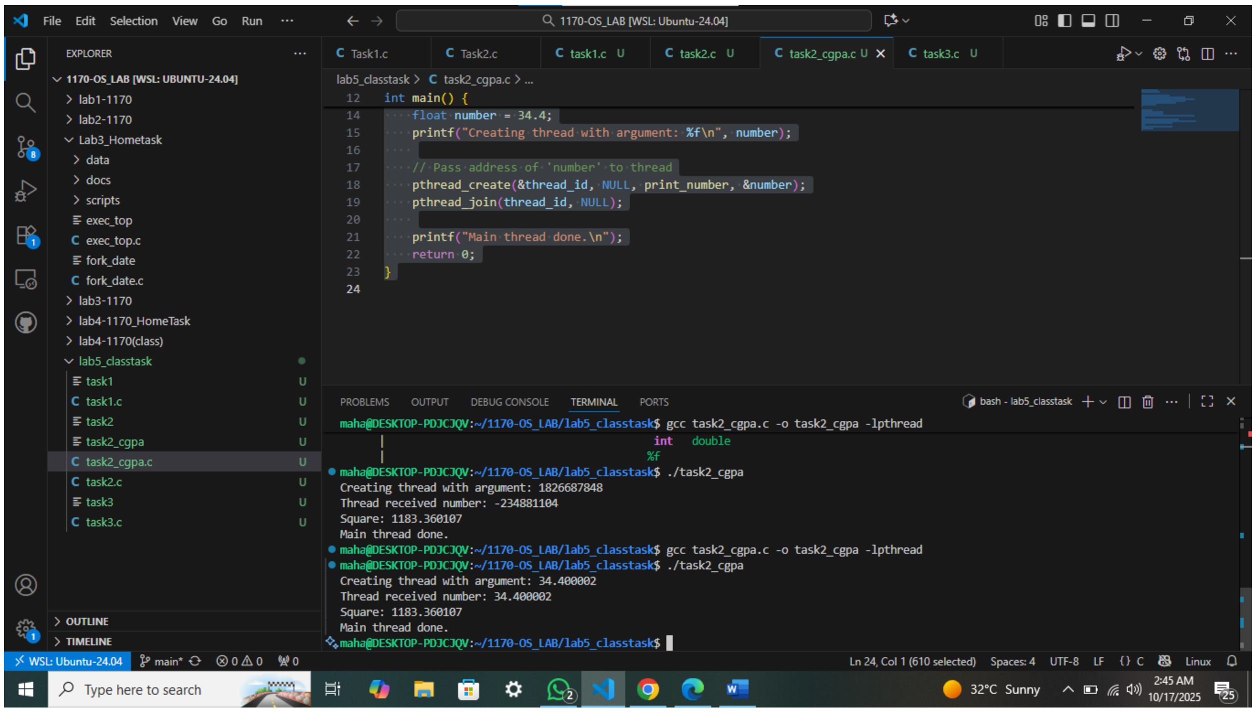
Task 2.2

Double CGPA using thread

Code:

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

void* print_number(void* arg) {
    // We know that we've passed a float pointer
    float num = *(float*)arg; // Cast void* back to float*
    printf("Thread received number: %f\n", num);
    printf("Square: %f\n", num * num);
    return NULL;
}

int main() {
    pthread_t thread_id;
    float number = 34.4;
    printf("Creating thread with argument: %f\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;
}
```

Terminal:



```c
int main() {
    float number = 34.4;
    printf("Creating thread with argument: %f\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;
}
```

```
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ gcc task2_cgpa.c -o task2_cgpa -lpthread
      |                                         int   double
      |                                        %f
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ ./task2_cgpa
Creating thread with argument: 1826687848
Thread received number: -234881104
Square: 1183.360107
Main thread done.
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ gcc task2_cgpa.c -o task2_cgpa -lpthread
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ ./task2_cgpa
Creating thread with argument: 34.400002
Thread received number: 34.400002
Square: 1183.360107
Main thread done.
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$
```

Program 3: Passing Multiple Data

Code:

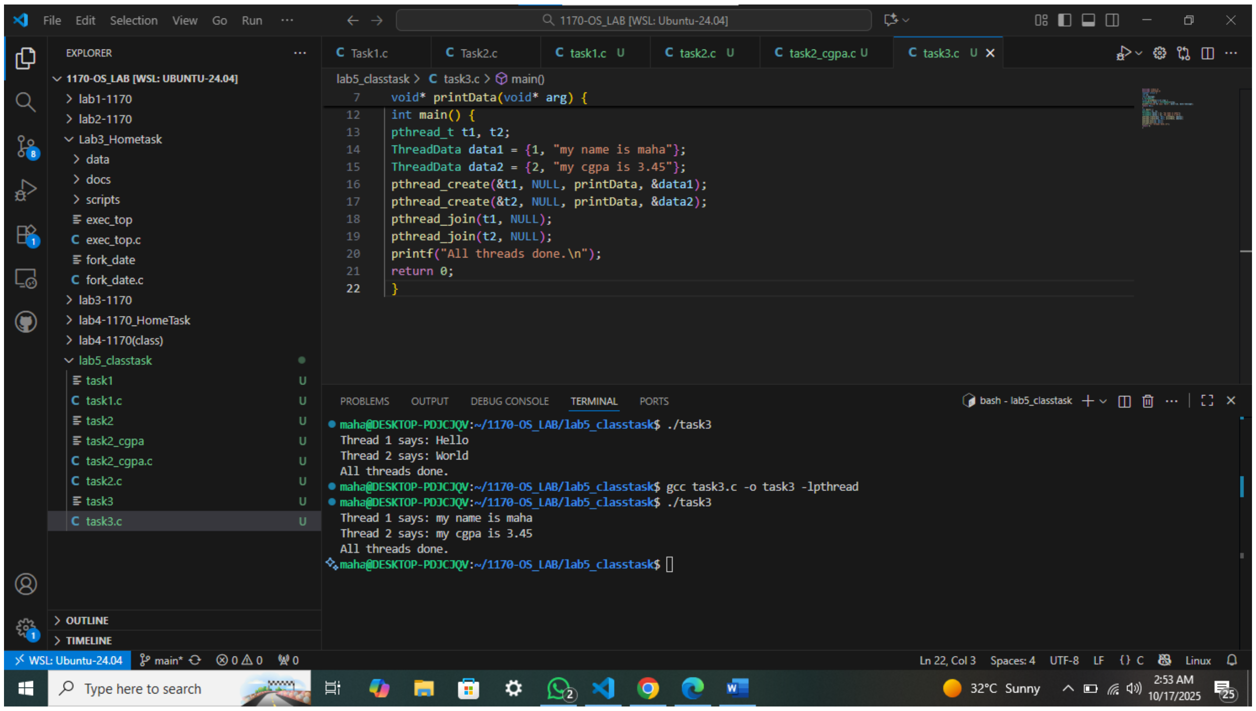
```c
#include <stdio.h>

#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, "my name is maha"};

ThreadData data2 = {2, "my cgpa is 3.45"};

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");

return 0;

}
```

Execution commands:

Compile and run:

```
gcc thread3.c -o thread3 -lpthread
./thread3
```

Terminal:



```
void* printData(void* arg) {
int main() {
pthread_t t1, t2;
ThreadData data1 = {1, "my name is maha"};
ThreadData data2 = {2, "my cgpa is 3.45"};
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");
return 0;
}
```

```
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ ./task3
Thread 1 says: Hello
Thread 2 says: World
All threads done.
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ gcc task3.c -o task3 -lpthread
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ ./task3
Thread 1 says: my name is maha
Thread 2 says: my cgpa is 3.45
All threads done.
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$
```

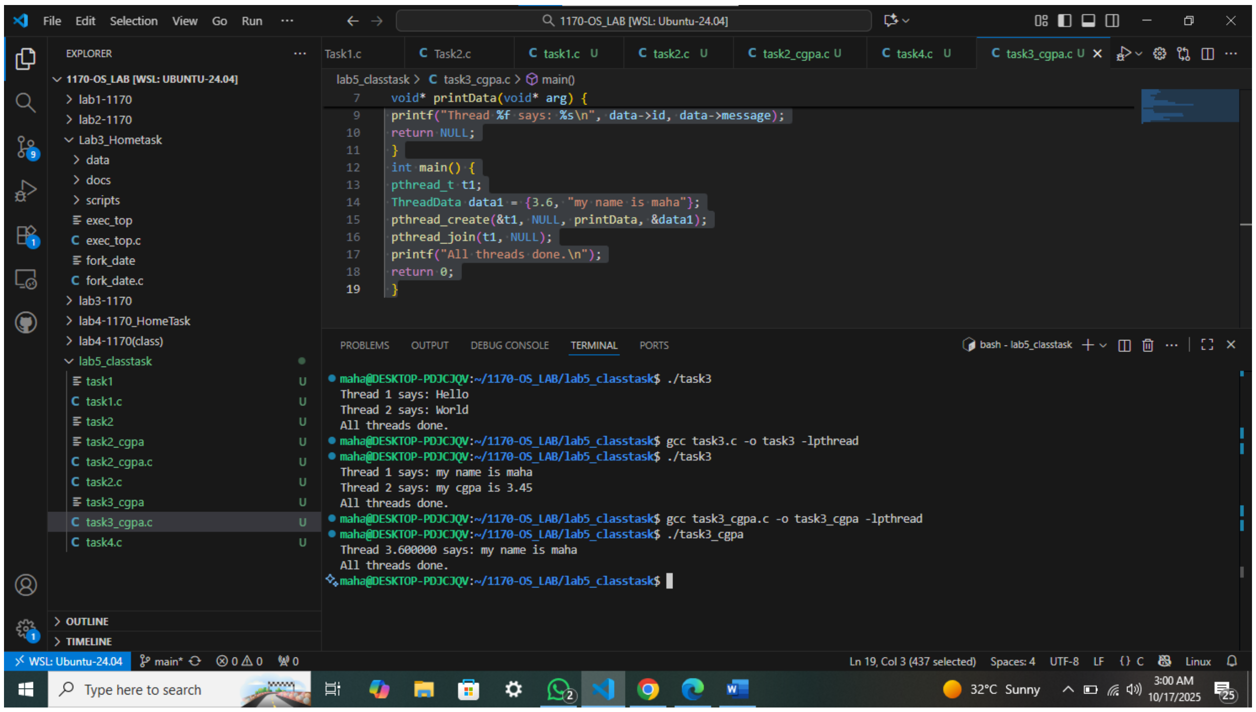
Task 3.2

Print your cgpa and name string in t1

Code:

```
#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 = {3.6, "my name is maha"};
pthread_create(&t1, NULL, printData, &data1);
pthread_join(t1, NULL);
printf("All threads done.\n");
return 0;
}
```

Terminal:



```c
void* printData(void* arg) {
printf("Thread %f says: %s\n", data->id, data->message);
return NULL;
}
int main() {
pthread_t t1;
ThreadData data1 = {3.6, "my name is maha"};
pthread_create(&t1, NULL, printData, &data1);
pthread_join(t1, NULL);
printf("All threads done.\n");
return 0;
}
```

```
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ ./task3
Thread 1 says: Hello
Thread 2 says: World
All threads done.
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ gcc task3.c -o task3 -lpthread
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ ./task3
Thread 1 says: my name is maha
Thread 2 says: my cgpa is 3.45
All threads done.
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ gcc task3_cgpa.c -o task3_cgpa -lpthread
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ ./task3_cgpa
Thread 3.600000 says: my name is maha
All threads done.
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$
```

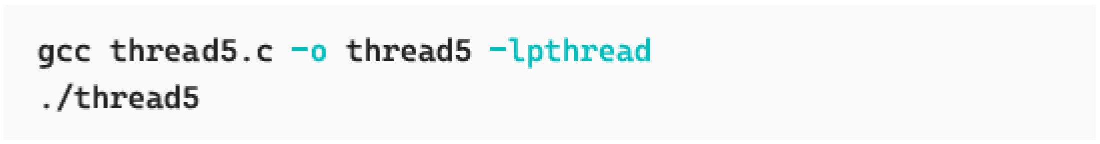
Program 4: Thread Return Values

Objective: Get return values from threads.

Code:

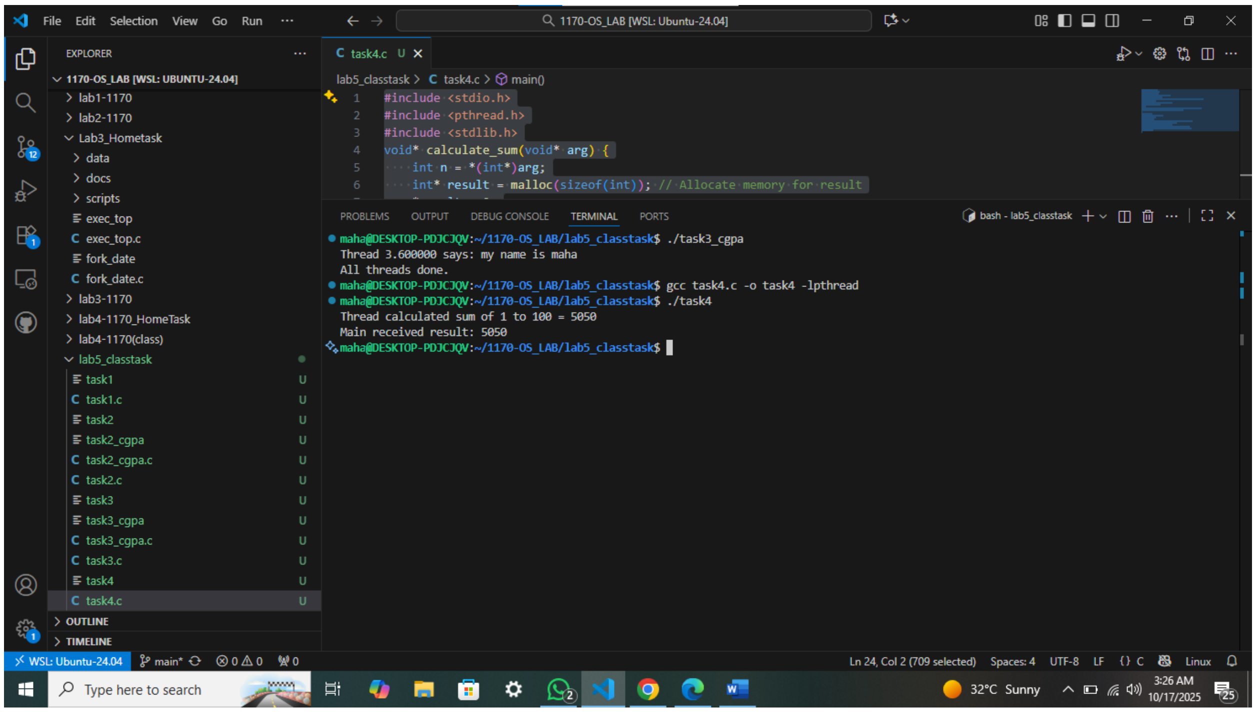
```c
#include <stdio.h>

#include <pthread.h>

#include <stdlib.h>

void* calculate_sum(void* arg) {

    int n = *(int*)arg;

    int* result = malloc(sizeof(int)); // Allocate memory for result

    *result = 0;

    for (int i = 1; i <= n; i++) {

        *result += i;

    }

    printf("Thread calculated sum of 1 to %d = %d\n", n, *result);

    return (void*)result; // Return the result
}
int main() {
    pthread_t thread_id;
    int n = 100;
    void* sum;
    pthread_create(&thread_id, NULL, calculate_sum, &n);
    // Get the return value from thread
    pthread_join(thread_id, &sum);
    printf("Main received result: %d\n", *(int*)sum);
    free(sum); // Don't forget to free allocated memory
    return 0;
}
```

Execution command:



```
gcc thread5.c -o thread5 -lpthread
./thread5
```

Terminal:



```c
#include <stdio.h>
#include <pthread.h>
#include <stdlib.h>
void* calculate_sum(void* arg) {
    int n = *(int*)arg;
    int* result = malloc(sizeof(int)); // Allocate memory for result
```

```
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ ./task3_cgpa
Thread 3.600000 says: my name is maha
All threads done.
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ gcc task4.c -o task4 -lpthread
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ ./task4
Thread calculated sum of 1 to 100 = 5050
Main received result: 5050
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ 
```

Program 1: Creating and Running Multiple Threads

Objective:

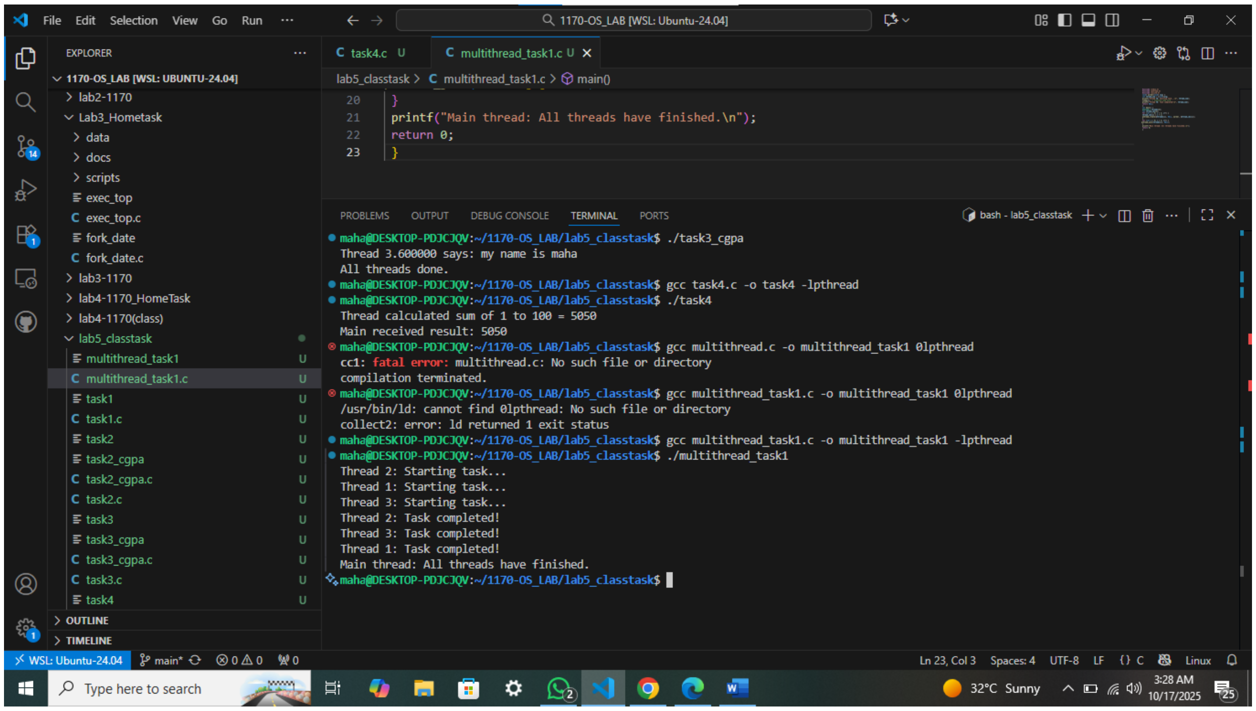
Create multiple threads that execute independently and print messages concurrently.

Code:

```c
#include <stdio.h>

#include <pthread.h>

#include <unistd.h>

void* worker(void* arg) {

int thread_num = *(int*)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;

}
```

Terminal:



```
}
printf("Main thread: All threads have finished.\n");
return 0;
}
```

```
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ ./task3_cgpa
Thread 3.600000 says: my name is maha
All threads done.
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ gcc task4.c -o task4 -lpthread
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ ./task4
Thread calculated sum of 1 to 100 = 5050
Main received result: 5050
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ gcc multithread.c -o multithread_task1 0lpthread
cc1: fatal error: multithread.c: No such file or directory
compilation terminated.
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ gcc multithread_task1.c -o multithread_task1 0lpthread
/usr/bin/ld: cannot find 0lpthread: No such file or directory
collect2: error: ld returned 1 exit status
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ gcc multithread_task1.c -o multithread_task1 -lpthread
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ ./multithread_task1
Thread 2: Starting task...
Thread 1: Starting task...
Thread 3: Starting task...
Thread 2: Task completed!
Thread 3: Task completed!
Thread 1: Task completed!
Main thread: All threads have finished.
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$
```

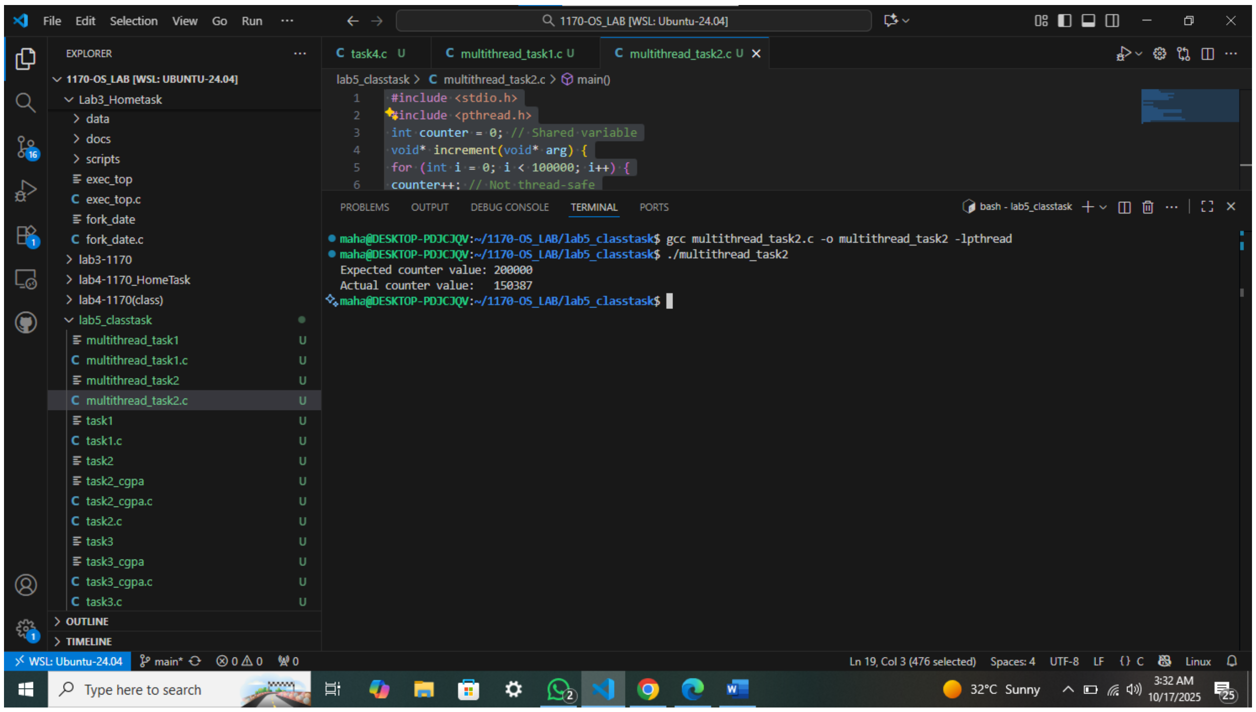
Program 2: Demonstrating a Race Condition

Objective: What happens when multiple threads modify a shared variable **without synchronization**.

Code:

```
 #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;

 }
```

Terminal:



```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
```

```
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ gcc multithread_task2.c -o multithread_task2 -lpthread
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$ ./multithread_task2
Expected counter value: 200000
Actual counter value:   150387
maha@DESKTOP-PDJCJQV:~/1170-OS_LAB/lab5_classtask$
```