



National Textile University

Department of Computer Science

Subject:

Operating Systems

Submitted To:

Sir Nasir Mehmood

Submitted By:

Alishba Riasat

Registration No:

23-NTU-CS-1135

Lab No:

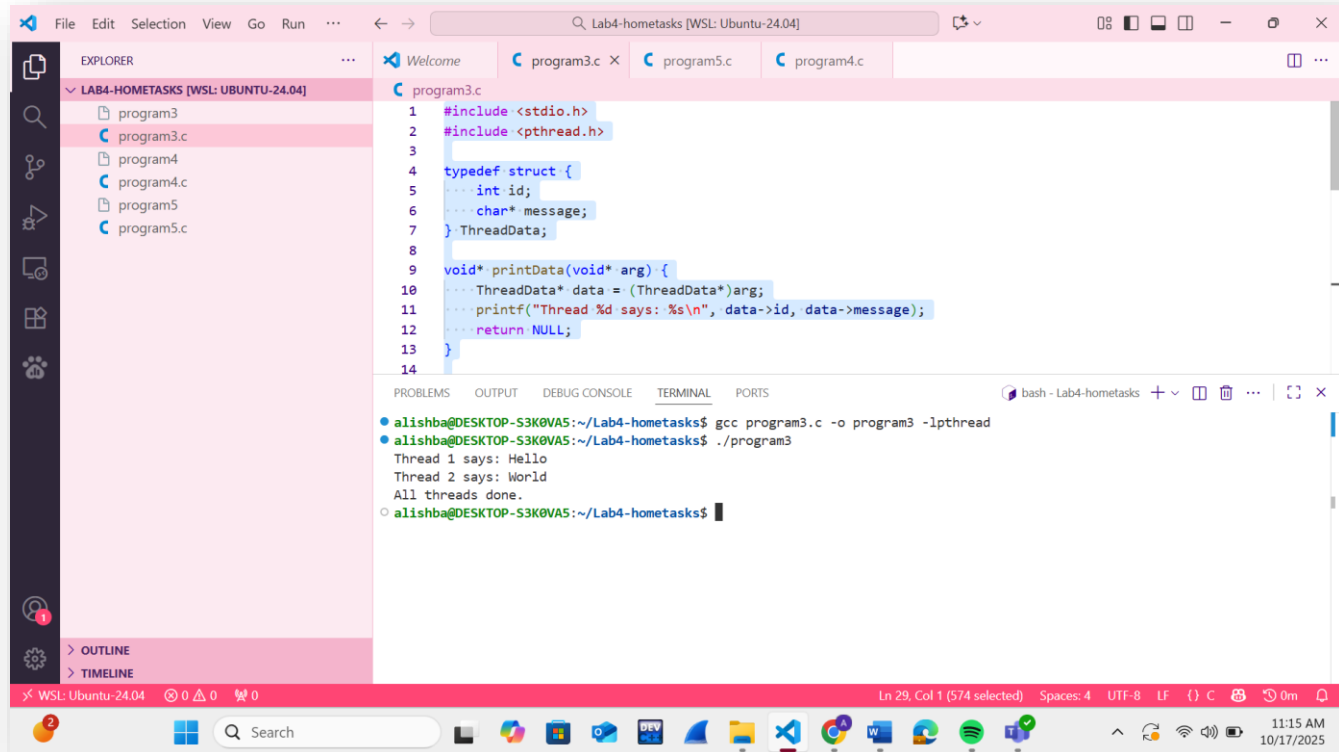
4-Hometasks

Semester:

5th

3. C Programs with Threads

Program 3: Passing Multiple Data



The screenshot displays the Visual Studio Code interface with a C program named `program3.c` open. The Explorer panel on the left shows the project structure under `LAB4-HOMETASKS [WSL: UBUNTU-24.04]`, including files `program3`, `program3.c`, `program4`, `program4.c`, `program5`, and `program5.c`. The main editor window shows the code for `program3.c`, which includes `<stdio.h>` and `<pthread.h>`. It defines a `ThreadData` struct with `int id` and `char* message`, and implements a `printData` function that prints the thread's ID and message. The terminal at the bottom shows the compilation and execution of the program, resulting in the output: "Thread 1 says: Hello", "Thread 2 says: World", and "All threads done."

```
1 #include <stdio.h>
2 #include <pthread.h>
3
4 typedef struct {
5     int id;
6     char* message;
7 } ThreadData;
8
9 void* printData(void* arg) {
10     ThreadData* data = (ThreadData*)arg;
11     printf("Thread %d says: %s\n", data->id, data->message);
12     return NULL;
13 }
14
```

```
alishba@DESKTOP-S3K0VA5:~/Lab4-hometasks$ gcc program3.c -o program3 -lpthread
alishba@DESKTOP-S3K0VA5:~/Lab4-hometasks$ ./program3
Thread 1 says: Hello
Thread 2 says: World
All threads done.
alishba@DESKTOP-S3K0VA5:~/Lab4-hometasks$
```

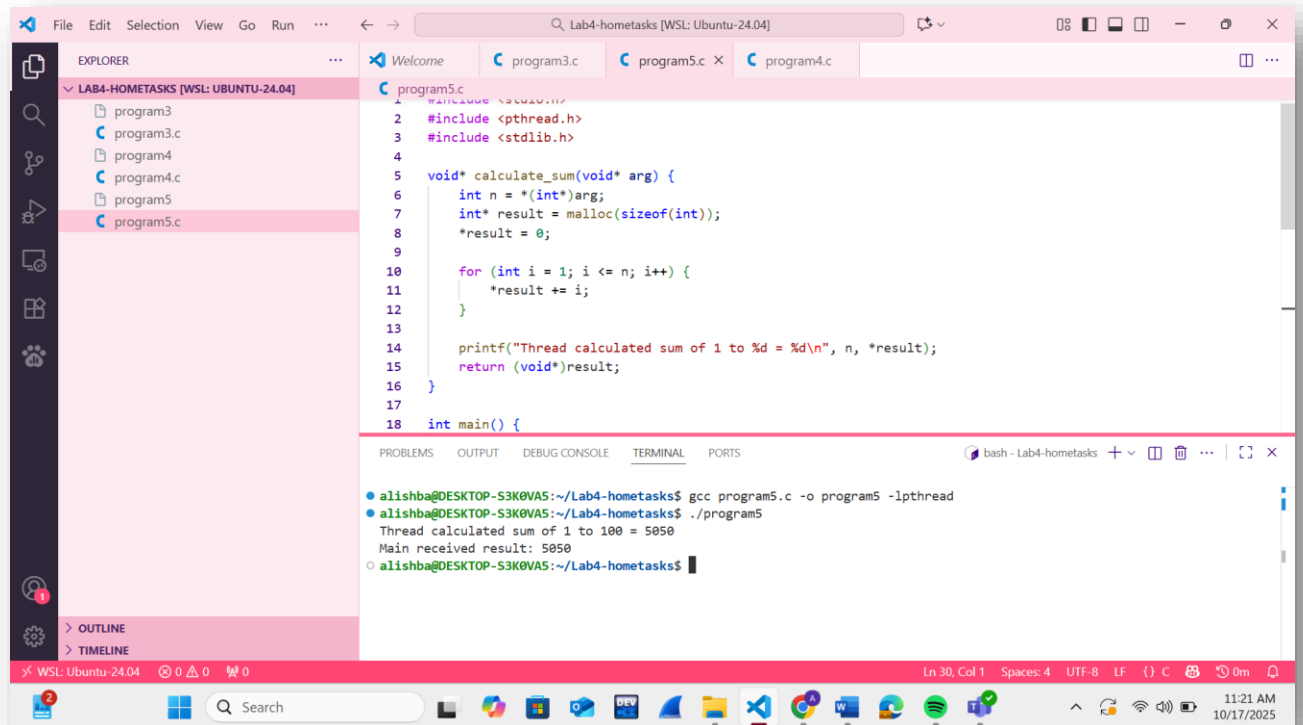
Program 4: Multiple threads

The screenshot shows the Visual Studio Code editor with the 'LAB4-HOMETASKS [WSL: UBUNTU-24.04]' workspace. The Explorer sidebar on the left lists files: program3, program3.c, program4, program4.c, program5, and program5.c. The main editor displays the code for 'program4.c', which includes `<stdio.h>`, `<pthread.h>`, and `<unistd.h>`. It defines a `worker_thread` function. The bottom panel shows the TERMINAL output, which logs the creation and completion of five threads. The command `gcc program4.c -o program4 -lpthread` was executed, followed by `./program4`. The output shows the main thread creating five worker threads, each starting work and then completing. The main thread then reports that each worker thread has finished, and finally states 'All threads completed!'. The status bar at the bottom indicates the current position is line 31, column 1, with 4 spaces, UTF-8 encoding, and LF line endings. The system clock shows 11:20 AM on 10/17/2025.

```
1 #include <stdio.h>
2 #include <pthread.h>
3 #include <unistd.h>
4
5 void* worker_thread(void* arg) {
6     int id = *(int*)arg;
7     printf("Thread %d: Starting work...\n", id);
8     sleep(1);
9     printf("Thread %d: Work completed!\n", id);
10 }
11
12 int main() {
13     pthread_t threads[5];
14     int args[5] = {1, 2, 3, 4, 5};
15     printf("Main: Creating thread 1\n");
16     pthread_create(&threads[0], NULL, worker_thread, (void*)&args[0]);
17     printf("Main: Creating thread 2\n");
18     pthread_create(&threads[1], NULL, worker_thread, (void*)&args[1]);
19     printf("Main: Creating thread 3\n");
20     pthread_create(&threads[2], NULL, worker_thread, (void*)&args[2]);
21     printf("Main: Creating thread 4\n");
22     pthread_create(&threads[3], NULL, worker_thread, (void*)&args[3]);
23     printf("Main: Creating thread 5\n");
24     pthread_create(&threads[4], NULL, worker_thread, (void*)&args[4]);
25     printf("Thread 1: Starting work...\n");
26     printf("Thread 2: Starting work...\n");
27     printf("Thread 3: Starting work...\n");
28     printf("Thread 4: Starting work...\n");
29     printf("Thread 5: Starting work...\n");
30     pthread_join(threads[0], NULL);
31     pthread_join(threads[1], NULL);
32     pthread_join(threads[2], NULL);
33     pthread_join(threads[3], NULL);
34     pthread_join(threads[4], NULL);
35     printf("Main: Thread 1 has finished\n");
36     printf("Main: Thread 2 has finished\n");
37     printf("Main: Thread 3 has finished\n");
38     printf("Main: Thread 4 has finished\n");
39     printf("Main: Thread 5 has finished\n");
40     printf("All threads completed!\n");
41     return 0;
42 }
```

```
alishba@DESKTOP-S3K0VA5:~/Lab4-hometasks$ gcc program4.c -o program4 -lpthread
alishba@DESKTOP-S3K0VA5:~/Lab4-hometasks$ ./program4
Main: Creating thread 1
Main: Creating thread 2
Thread 1: Starting work...
Main: Creating thread 3
Thread 2: Starting work...
Main: Creating thread 4
Thread 3: Starting work...
Main: Creating thread 5
Thread 4: Starting work...
Thread 5: Starting work...
Thread 1: Work completed!
Thread 3: Work completed!
Thread 2: Work completed!
Thread 5: Work completed!
Main: Thread 1 has finished
Main: Thread 2 has finished
Main: Thread 3 has finished
Thread 4: Work completed!
Main: Thread 4 has finished
Main: Thread 5 has finished
All threads completed!
alishba@DESKTOP-S3K0VA5:~/Lab4-hometasks$
```

Program 5: Thread Return Values



The screenshot displays the Visual Studio Code interface with the file explorer on the left showing a project named 'LAB4-HOMETASKS [WSL: UBUNTU-24.04]'. The file explorer lists files: program3, program3.c, program4, program4.c, program5, and program5.c. The main editor window shows the source code for 'program5.c'.

```
1 #include <stdio.h>
2 #include <pthread.h>
3 #include <stdlib.h>
4
5 void* calculate_sum(void* arg) {
6     int n = *(int*)arg;
7     int* result = malloc(sizeof(int));
8     *result = 0;
9
10    for (int i = 1; i <= n; i++) {
11        *result += i;
12    }
13
14    printf("Thread calculated sum of 1 to %d = %d\n", n, *result);
15    return (void*)result;
16 }
17
18 int main() {
```

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

```
alishba@DESKTOP-S3K0VA5:~/Lab4-hometasks$ gcc program5.c -o program5 -lpthread
alishba@DESKTOP-S3K0VA5:~/Lab4-hometasks$ ./program5
Thread calculated sum of 1 to 100 = 5050
Main received result: 5050
alishba@DESKTOP-S3K0VA5:~/Lab4-hometasks$
```

The status bar at the bottom indicates the current file is 'program5.c' at line 30, column 1, with 4 spaces, UTF-8 encoding, and LF line endings. The system clock shows 11:21 AM on 10/17/2025.

Exercise 1:

The screenshot shows the Visual Studio Code editor with the file explorer on the left displaying the 'LAB4-HOMETASKS' directory. The main editor window shows the code for 'exercise1.c'. The code includes `<stdio.h>` and `<pthread.h>`. It defines a `print_message` function that takes a pointer to an integer and prints a message. The `main` function creates an array of three threads, each with its own ID, and calls `pthread_create` to start them. The threads print messages and then return NULL. The terminal output shows the compilation and execution of the program, resulting in three threads printing their IDs.

```
1 #include <stdio.h>
2 #include <pthread.h>
3
4 void* print_message(void* arg) {
5     int id = *(int*)arg;
6     printf("Thread %d: Hello from thread %d!\n", id, id);
7     return NULL;
8 }
9
10 int main() {
11     pthread_t threads[3];
12     int ids[3];
13
14     for (int i = 0; i < 3; i++) {
15         ids[i] = i + 1;
16         pthread_create(&threads[i], NULL, print_message, &ids[i]);
17     }
18
19     for (int i = 0; i < 3; i++) {
20         pthread_join(threads[i], NULL);
21     }
22
23     printf("Main thread: All threads completed.\n");
24     return 0;
25 }
```

Terminal Output:

```
alishba@DESKTOP-S3K0VA5:~/Lab4-hometasks$ gcc exercise1.c -o exercise1 -lpthread
alishba@DESKTOP-S3K0VA5:~/Lab4-hometasks$ ./exercise1
Thread 1: Hello from thread 1!
Thread 2: Hello from thread 2!
Thread 3: Hello from thread 3!
Main thread: All threads completed.
alishba@DESKTOP-S3K0VA5:~/Lab4-hometasks$
```

Exercise 2:

The screenshot shows the Visual Studio Code editor with the file explorer on the left displaying the 'LAB4-HOMETASKS' directory. The main editor window shows the code for 'exercise2.c'. The code includes `<stdio.h>`, `<pthread.h>`, and `<stdlib.h>`. It defines a `check_prime` function that takes a pointer to an integer and checks if it is a prime number. The `main` function creates an array of three threads, each with its own ID, and calls `pthread_create` to start them. The threads check if the number is prime and print the result. The terminal output shows the compilation and execution of the program, resulting in three threads checking the number 5 and printing '5 is a prime number'.

```
1 #include <stdio.h>
2 #include <pthread.h>
3 #include <stdlib.h>
4
5 void* check_prime(void* arg) {
6     int n = *(int*)arg;
7     int* result = malloc(sizeof(int));
8     *result = 1;
9
10    if (n <= 1) *result = 0;
11    else {
12        for (int i = 2; i * i <= n; i++) {
13            if (n % i == 0) {
14                *result = 0;
15                break;
16            }
17        }
18    }
19
20    return result;
21 }
```

Terminal Output:

```
alishba@DESKTOP-S3K0VA5:~/Lab4-hometasks$ gcc exercise2.c -o exercise2 -lpthread
alishba@DESKTOP-S3K0VA5:~/Lab4-hometasks$ ./exercise2
Enter a number: 5
5 is a prime number.
alishba@DESKTOP-S3K0VA5:~/Lab4-hometasks$
```

