



National Textile University

Department of Computer Science

Subject:

Operating System

Submitted to:

Sir Naseer

Submitted by:

Ayema

Reg number:

23-NTU-CS-1142

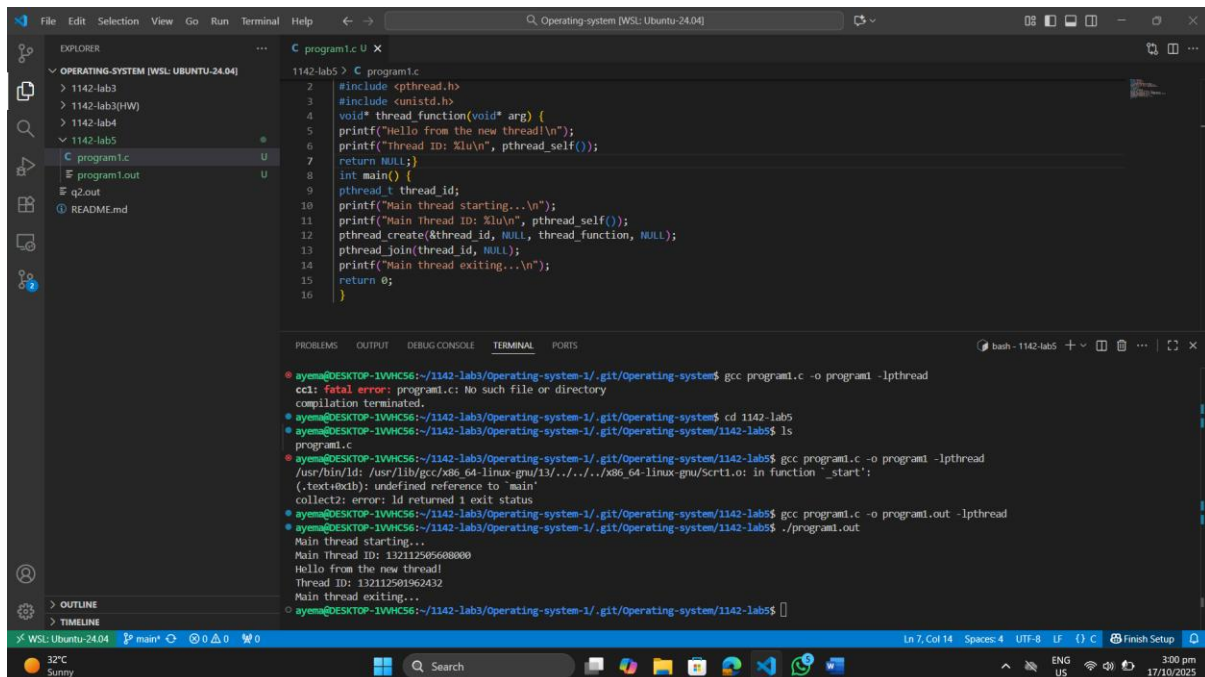
Lab no. :

05

Semester: 5Th

Program 1: Creating a Simple Thread

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

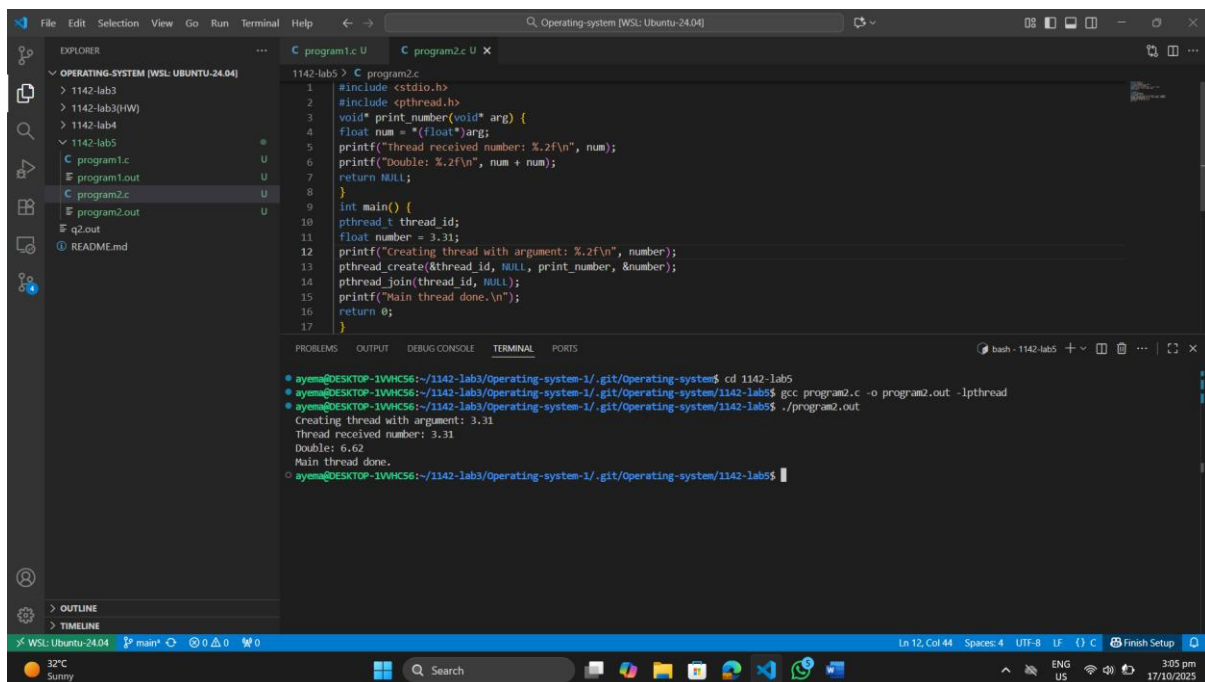


```
WSL: Ubuntu-24.04
File Edit Selection View Go Run Terminal Help
1142-lab5 > C program1.c
1  #include <stdio.h>
2  #include <pthread.h>
3  #include <unistd.h>
4  void* thread_function(void* arg) {
5      printf("Hello from the new thread!\n");
6      printf("Thread ID: %lu\n", pthread_self());
7      return NULL;}
8  int main() {
9      pthread_t thread_id;
10     printf("Main thread starting...\n");
11     printf("Main Thread ID: %lu\n", pthread_self());
12     pthread_create(&thread_id, NULL, thread_function, NULL);
13     pthread_join(thread_id, NULL);
14     printf("Main thread exiting...\n");
15     return 0;
16 }

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
ayema@DESKTOP-1VHHC56:~/1142-lab3/Operating-system-1/.git/Operating-system-1$ gcc program1.c -o program1 -lpthread
cc1: fatal error: program1.c: No such file or directory
compilation terminated.
ayema@DESKTOP-1VHHC56:~/1142-lab3/Operating-system-1/.git/Operating-system-1$ cd 1142-lab5
ayema@DESKTOP-1VHHC56:~/1142-lab3/Operating-system-1/.git/Operating-system-1/1142-lab5$ ls
program1.c
ayema@DESKTOP-1VHHC56:~/1142-lab3/Operating-system-1/.git/Operating-system-1/1142-lab5$ gcc program1.c -o program1 -lpthread
/usr/bin/ld: /usr/lib/gcc/x86_64-linux-gnu/13/../../../../x86_64-linux-gnu/libcrt1.o: in function '_start':
(.text+0x1b): undefined reference to 'main'
collect2: error: ld returned 1 exit status
ayema@DESKTOP-1VHHC56:~/1142-lab3/Operating-system-1/.git/Operating-system-1/1142-lab5$ gcc program1.c -o program1.out -lpthread
Main thread starting...
Main Thread ID: 13211250608000
Hello from the new thread!
Thread ID: 132112501962432
Main thread exiting...
ayema@DESKTOP-1VHHC56:~/1142-lab3/Operating-system-1/.git/Operating-system-1/1142-lab5$
```

Program 2: Passing Arguments to Threads

```
1  #include <stdio.h>
2  #include <pthread.h>
3  void* print_number(void* arg) {
4  float num = *(float*)arg;
5  printf("Thread received number: %.2f\n", num);
6  printf("Double: %.2f\n", num + num);
7  return NULL;
8  }
9  int main() {
10 pthread_t thread_id;
11 float number = 3.31;
12 printf("Creating thread with argument: %.2f\n", number);
13 pthread_create(&thread_id, NULL, print_number, &number);
14 pthread_join(thread_id, NULL);
15 printf("Main thread done.\n");
16 return 0;
17 }
```




The screenshot displays the Visual Studio Code interface with the C program code in the editor and its execution output in the terminal. The code is the same as shown in the previous block. The terminal output shows the following sequence of events:

```
ayema@DESKTOP-1VHKS6:~/1142-lab3/Operating-system-1/.git/Operating-system$ cd 1142-lab5
ayema@DESKTOP-1VHKS6:~/1142-lab3/Operating-system-1/.git/Operating-system/1142-lab5$ gcc program2.c -o program2.out -lpthread
ayema@DESKTOP-1VHKS6:~/1142-lab3/Operating-system-1/.git/Operating-system/1142-lab5$ ./program2.out
Creating thread with argument: 3.31
Thread received number: 3.31
Double: 6.62
Main thread done.
```

The status bar at the bottom indicates the file is 'main.c' in the 'main' namespace, with 0 errors and 0 warnings. The system tray shows a temperature of 32°C and the date/time as 3:05 pm on 17/10/2025.

Program 3: Passing Multiple Data:



```
1  #include <stdio.h>
2  #include <pthread.h>
3  typedef struct {
4      char* message;
5      float value;
6  } ThreadData;
7  void* printData(void* arg) {
8      ThreadData* data = (ThreadData*)arg;
9      printf("%s\n", data->message);
10     printf("%.2f\n", data->value);
11     return NULL;
12 }
13 int main() {
14     pthread_t t1;
15     ThreadData data1 = {"Ayema", 3.31};
16     pthread_create(&t1, NULL, printData, &data1);
17     pthread_join(t1, NULL);
18     return 0;
19 }
20
```

The screenshot shows the Visual Studio Code editor with a C program named `program3.c` open. The program defines a `ThreadData` struct with `message` and `value` fields, and a `printData` function that prints these fields. The `main` function creates a thread `t1` with `data1` containing the string "Ayema" and the value 3.31, then joins the thread. The terminal output shows the command to compile and run the program, followed by the output "Ayema" and "3.31".

```
1 #include <stdio.h>
2 #include <pthread.h>
3 typedef struct {
4     char* message;
5     float value;
6 } ThreadData;
7 void* printData(void* arg) {
8     ThreadData* data = (ThreadData*)arg;
9     printf("%s\n", data->message);
10    printf("%.2f\n", data->value);
11    return NULL;
12 }
13 int main() {
14     pthread_t t1;
15     ThreadData data1 = {"Ayema", 3.31};
16     pthread_create(&t1, NULL, printData, &data1);
17     pthread_join(t1, NULL);
18 }
```

```
ayema@DESKTOP-1VHKS6:~/1142-lab3/Operating-system-1/.git/Operating-system/1142-lab5$ cd 1142-lab5
ayema@DESKTOP-1VHKS6:~/1142-lab3/Operating-system-1/.git/Operating-system/1142-lab5$ gcc program3.c -o program3.out -lpthread
ayema@DESKTOP-1VHKS6:~/1142-lab3/Operating-system-1/.git/Operating-system/1142-lab5$ ./program3.out
Ayema
3.31
ayema@DESKTOP-1VHKS6:~/1142-lab3/Operating-system-1/.git/Operating-system/1142-lab5$
```

Program 4: Thread Return Values

The screenshot shows a terminal window with the code for Program 4. The program defines a `calculate_sum` function that calculates the sum of integers from 1 to `n` and returns the result. The `main` function creates a thread `thread_id` with `sum` as the argument, then joins the thread and prints the result.

```
1 #include <stdio.h>
2 #include <pthread.h>
3 #include <stdlib.h>
4 void* calculate_sum(void* arg) {
5     int n = *(int*)arg;
6     int* result = malloc(sizeof(int));
7     *result = 0;
8     for (int i = 1; i <= n; i++) {
9         *result += i;
10    }
11    printf("Thread calculated sum of 1 to %d = %d\n", n, *result);
12    return (void*)result;
13 }
14 int main() {
15     pthread_t thread_id;
16     int n = 100;
17     void* sum;
18     pthread_create(&thread_id, NULL, calculate_sum, &n);
19     pthread_join(thread_id, &sum);
20     printf("Main received result: %d\n", *(int*)sum);
21     free(sum);
22     return 0;
23 }
```

The screenshot shows the Visual Studio Code editor with a C program named `program4.c` open. The program uses `pthread.h` to create and manage multiple threads. The code includes headers for `stdio.h`, `pthread.h`, and `unistd.h`. It defines a `calculate_sum` function that calculates the sum of integers from 1 to `n` in a separate thread. The `main` function creates three threads, each calling `calculate_sum` with a different value of `n` (1, 100, and 5000). The threads are joined back to the main thread, and the results are printed.

```
1 #include <stdio.h>
2 #include <pthread.h>
3 #include <unistd.h>
4 void* calculate_sum(void* arg) {
5     int n = *(int*)arg;
6     int* result = malloc(sizeof(int));
7     *result = 0;
8     for (int i = 1; i <= n; i++) {
9         *result += i;
10    }
11    printf("Thread calculated sum of 1 to %d = %d\n", n, *result);
12    return (void*)result;
13 }
14 int main() {
15     pthread_t thread_id;
16     int n = 1000;
17     void*
```

The terminal output shows the execution of the program. It displays the thread IDs, the calculated sums for each thread, and the final result from the main thread.

```
ayma@DESKTOP-1VHCS6:~/1142-lab3/Operating-system-1/.git/Operating-system$ cd 1142-lab5
ayma@DESKTOP-1VHCS6:~/1142-lab3/Operating-system-1/.git/Operating-system/1142-lab5$ gcc program4.c -o program4.out -lpthread
ayma@DESKTOP-1VHCS6:~/1142-lab3/Operating-system-1/.git/Operating-system/1142-lab5$ ./program4.out
Thread calculated sum of 1 to 100 = 5050
Thread calculated sum of 1 to 5000 = 12502500
Main received result: 5050
ayma@DESKTOP-1VHCS6:~/1142-lab3/Operating-system-1/.git/Operating-system/1142-lab5$
```

Program 4.1: Creating and Running Multiple Threads

```
1 #include <stdio.h>
2 #include <pthread.h>
3 #include <unistd.h>
4 void* worker(void* arg) {
5     int thread_num = *(int*)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 }
24
```

The screenshot shows the Visual Studio Code editor with a C program named `program4.1.c` open. The program uses `pthread` to create three threads that each print a message and then return `NULL`. The `main` function waits for all threads to finish before printing a completion message. The terminal output shows the program being compiled and executed, with the expected output messages appearing in the order: Thread 1, Thread 3, Thread 2, followed by the completion messages and the final 'All threads have finished' message.

```
1 #include <stdio.h>
2 #include <pthread.h>
3 #include <unistd.h>
4 void* worker(void* arg) {
5     int thread_num = *(int*)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 }
```

```
ayema@DESKTOP-1VHCS6:~/1142-lab3/Operating-system-1/.git/Operating-system$ cd 1142-lab3
ayema@DESKTOP-1VHCS6:~/1142-lab3/Operating-system-1/.git/Operating-system/1142-lab3$ gcc program4.1.c -o program4.1.out -lpthread
ayema@DESKTOP-1VHCS6:~/1142-lab3/Operating-system-1/.git/Operating-system/1142-lab3$ ./program4.1.out
Thread 1: Starting task...
Thread 3: Starting task...
Thread 2: Starting task...
Thread 1: Task completed!
Thread 3: Task completed!
Thread 2: Task completed!
Main thread: All threads have finished.
ayema@DESKTOP-1VHCS6:~/1142-lab3/Operating-system-1/.git/Operating-system/1142-lab3$
```

Program 4.2: Demonstrating a Race Condition

```
1  #include <stdio.h>
2  #include <pthread.h>
3  int counter = 0;
4  void* increment(void* arg) {
5      for (int i = 0; i < 100000; i++) {
6          counter++;
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;}
```

