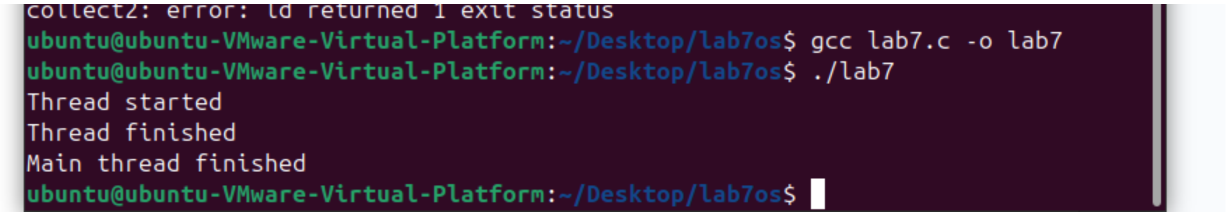
NAME MONEEB TOOR
ROLL NMBR 221048
LAB 07
AIR UNIVERSITY ISLAMABAD

TASK1:

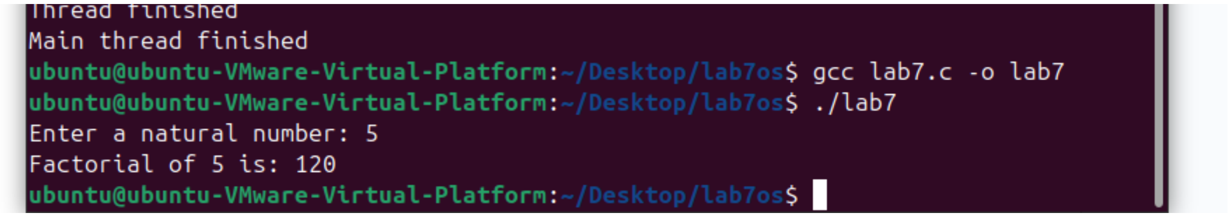
```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
#include <unistd.h>


void* thread_function(void* arg) {
        printf("Thread started\n");
        sleep(5); // Simulate some work
        printf("Thread finished\n");
        return NULL;
}

int main() {
        pthread_t thread_id;
        pthread_attr_t attr;


        pthread_attr_init(&attr);
        pthread_attr_setdetachstate(&attr, PTHREAD_CREATE_JOINABLE);


        pthread_create(&thread_id, &attr, thread_function, NULL);


        pthread_join(thread_id, NULL);


        pthread_attr_destroy(&attr);

        printf("Main thread finished\n");

        return 0;
}
```



```
collect2: error: ld returned 1 exit status
ubuntu@ubuntu-VMware-Virtual-Platform:~/Desktop/lab7os$ gcc lab7.c -o lab7
ubuntu@ubuntu-VMware-Virtual-Platform:~/Desktop/lab7os$ ./lab7
Thread started
Thread finished
Main thread finished
ubuntu@ubuntu-VMware-Virtual-Platform:~/Desktop/lab7os$ 
```

TASK 2:

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


long long factorial(int n) {
        if (n == 0 || n == 1) {
        return 1;
        } else {
        return n * factorial(n - 1);
        }
}


void* thread_function(void* arg) {
        int* num = (int*)arg;
        long long result = factorial(*num);
        printf("Factorial of %d is: %lld\n", *num, result);
        return NULL;
}

int main() {
        pthread_t thread;
        int n;

        printf("Enter a natural number: ");
        scanf("%d", &n);


        pthread_create(&thread, NULL, thread_function, &n);


        pthread_join(thread, NULL);

        return 0;
}
```



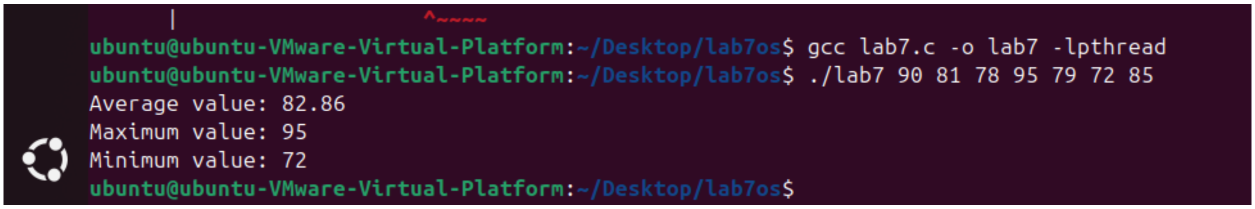
```
Thread finished
Main thread finished
ubuntu@ubuntu-VMware-Virtual-Platform:~/Desktop/lab7os$ gcc lab7.c -o lab7
ubuntu@ubuntu-VMware-Virtual-Platform:~/Desktop/lab7os$ ./lab7
Enter a natural number: 5
Factorial of 5 is: 120
ubuntu@ubuntu-VMware-Virtual-Platform:~/Desktop/lab7os$ 
```

TASK 3:

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

// Global variables to store the results
double average = 0.0;
int max_value = 0;
int min_value = 0;

// Function prototypes
void* calculate_average(void* arg);
void* calculate_max(void* arg);
void* calculate_min(void* arg);

int main(int argc, char* argv[]) {
        if (argc < 2) {
        fprintf(stderr, "Usage: %s number1 number2 ... numberN\n", argv[0]);
        return 1;
        }

        // Convert command-line arguments to integers
        int count = argc - 1;
        int* numbers = (int*)malloc(count * sizeof(int));
        for (int i = 0; i < count; i++) {
        numbers[i] = atoi(argv[i + 1]);
        }

        // Initialize global variables
        max_value = numbers[0];
        min_value = numbers[0];

        // Create threads
        pthread_t threads[3];
        pthread_create(&threads[0], NULL, calculate_average, (void*)numbers);
        pthread_create(&threads[1], NULL, calculate_max, (void*)numbers);
```

```
    pthread_create(&threads[2], NULL, calculate_min, (void*)numbers);

    // Wait for threads to finish
    for (int i = 0; i < 3; i++) {
    pthread_join(threads[i], NULL);
    }

    // Output the results
    printf("Average value: %.2f\n", average);
    printf("Maximum value: %d\n", max_value);
    printf("Minimum value: %d\n", min_value);

    // Free allocated memory
    free(numbers);

    return 0;
}

void* calculate_average(void* arg) {
    int* numbers = (int*)arg;
    int count = 0;
    double sum = 0.0;

    while (numbers[count] != 0) {
    sum += numbers[count];
    count++;
    }

    average = sum / count;
    pthread_exit(NULL);
}

void* calculate_max(void* arg) {
    int* numbers = (int*)arg;
    int count = 0;

    while (numbers[count] != 0) {
    if (numbers[count] > max_value) {
    max_value = numbers[count];
    }
    count++;
    }

    pthread_exit(NULL);
```

```
}

void* calculate_min(void* arg) {
        int* numbers = (int*)arg;
        int count = 0;

        while (numbers[count] != 0) {
        if (numbers[count] < min_value) {
        min_value = numbers[count];
        }
        count++;
        }

        pthread_exit(NULL);
}
```

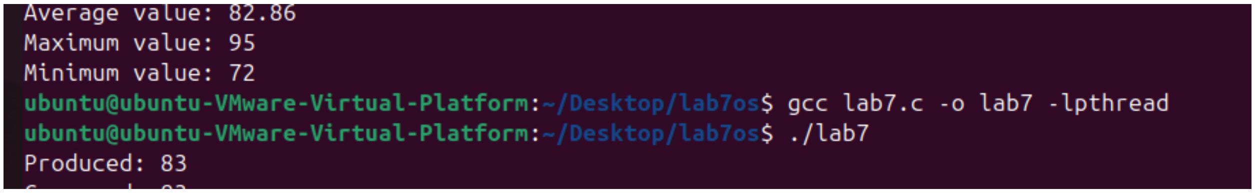


```
      |                    ^~~~~
ubuntu@ubuntu-VMware-Virtual-Platform:~/Desktop/lab7os$ gcc lab7.c -o lab7 -lpthread
ubuntu@ubuntu-VMware-Virtual-Platform:~/Desktop/lab7os$ ./lab7 90 81 78 95 79 72 85
Average value: 82.86
Maximum value: 95
Minimum value: 72
ubuntu@ubuntu-VMware-Virtual-Platform:~/Desktop/lab7os$
```

TASK 4:

```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
#include <semaphore.h>
#include <unistd.h>

#define BUFFER_SIZE 100
#define MAX_PRODUCE 100

int buffer[BUFFER_SIZE];
int in = 0;
int out = 0;

sem_t mutex;
sem_t full;
sem_t empty;

void* producer(void* arg) {
        int item;
```

```
    for (int i = 0; i < MAX_PRODUCE; i++) {
    item = rand() % 100; // Produce a random integer between 0 and 99

    sem_wait(&empty);
    sem_wait(&mutex);

    buffer[in] = item;
    printf("Produced: %d\n", item);
    in = (in + 1) % BUFFER_SIZE;

    sem_post(&mutex);
    sem_post(&full);

    sleep(1); // Simulate time taken to produce an item
    }
    pthread_exit(NULL);
}

void* consumer(void* arg) {
    int item;
    for (int i = 0; i < MAX_PRODUCE; i++) {
    sem_wait(&full);
    sem_wait(&mutex);

    item = buffer[out];
    printf("Consumed: %d\n", item);
    out = (out + 1) % BUFFER_SIZE;

    sem_post(&mutex);
    sem_post(&empty);

    sleep(1); // Simulate time taken to consume an item
    }
    pthread_exit(NULL);
}

int main() {
    pthread_t producer_thread, consumer_thread;

    // Initialize semaphores
    sem_init(&mutex, 0, 1);
    sem_init(&full, 0, 0);
    sem_init(&empty, 0, BUFFER_SIZE);
    // Create producer and consumer threads
    pthread_create(&producer_thread, NULL, producer, NULL);
    pthread_create(&consumer_thread, NULL, consumer, NULL);

    // Wait for threads to finish
    pthread_join(producer_thread, NULL);
    pthread_join(consumer_thread, NULL);

    // Destroy semaphores
    sem_destroy(&mutex);
    sem_destroy(&full);
    sem_destroy(&empty);

    return 0;
}
```



```
Average value: 82.86
Maximum value: 95
Minimum value: 72
ubuntu@ubuntu-VMware-Virtual-Platform:~/Desktop/lab7os$ gcc lab7.c -o lab7 -lpthread
ubuntu@ubuntu-VMware-Virtual-Platform:~/Desktop/lab7os$ ./lab7
Produced: 83
```