A logo of a company

AI-generated content may be incorrect.

**National Textile University**

**Department of Computer Science**

Subject: Operating System

Submitted to: Nasir Mahmood

Submitted by: Eemaan Fatima

Reg number:23-NTU-CS-1146

Assignment No.5

Semester:5th

Lab manual 05

**Task 1:**

 #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;

 }

A screenshot of a computer

AI-generated content may be incorrect.

Task 2:

 //Passing Arguments to Threads

 // Cgpa of a student and double it

 #include <stdio.h>

 #include <pthread.h>

 void\* print\_number(void\* arg) {

// We know that we've passed an integer pointer

 float num = \*(float\*)arg; // Cast void\* back to int\*

 printf("Thread received number: %f\n", num);

 printf("Square of cgpa: %f\n", num \* num);

 return NULL;

 }

 int main() {

 pthread\_t thread\_id;

float cgpa = 3.2;

 printf("Creating thread with argument: %f\n", cgpa);

 // Pass address of 'number' to thread

 pthread\_create(&thread\_id, NULL, print\_number, &cgpa);

 pthread\_join(thread\_id, NULL);

 printf("Main thread done.\n");

 return 0;

 }

A screenshot of a computer

AI-generated content may be incorrect.

Task 3:

//Passing Arguments to Threads

// Cgpa of a student

#include <stdio.h>

 #include <pthread.h>

 void\* print\_number(void\* arg) {

// We know that we've passed an integer pointer

 float num = \*(float\*)arg; // Cast void\* back to int\*

 printf("Thread received number: %f\n", num);

 printf("Double of cgpa: %f\n", num \* 2);

 return NULL;

 }

 int main() {

 pthread\_t thread\_id;

float cgpa = 3.2;

 printf("Creating thread with argument: %f\n", cgpa);

 // Pass address of 'number' to thread

 pthread\_create(&thread\_id, NULL, print\_number, &cgpa);

 pthread\_join(thread\_id, NULL);

 printf("Main thread done.\n");

 return 0;

 }

A screenshot of a computer

AI-generated content may be incorrect.

Task 4:

//PASING MULTIPLE ARGUMENTS DATA TYPES

//Cgpa of 2 people

#include <stdio.h>

 #include <pthread.h>

 typedef struct {

 float cgpa;

 char\* message;

 } ThreadData;

 void\* printData(void\* arg) {

 ThreadData\* data = (ThreadData\*)arg;

 printf("cgps %f is %s\n", data->cgpa, data->message);

 return NULL;

 }

 int main() {

 pthread\_t t1, t2;

 ThreadData data1 = {3.2, "My name is Eman"};

 ThreadData data2 = {3, "My name is Nimra"};

 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;

 }

A screenshot of a computer

AI-generated content may be incorrect.

Task 5:

 #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;

 }

A screenshot of a computer

AI-generated content may be incorrect.

Task 6:

#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, "Hello"};

    ThreadData data2 = {2, "World"};

    pthread\_create(&t1, NULL, printData, &data1);

    pthread\_create(&t2, NULL, printData, &data2);

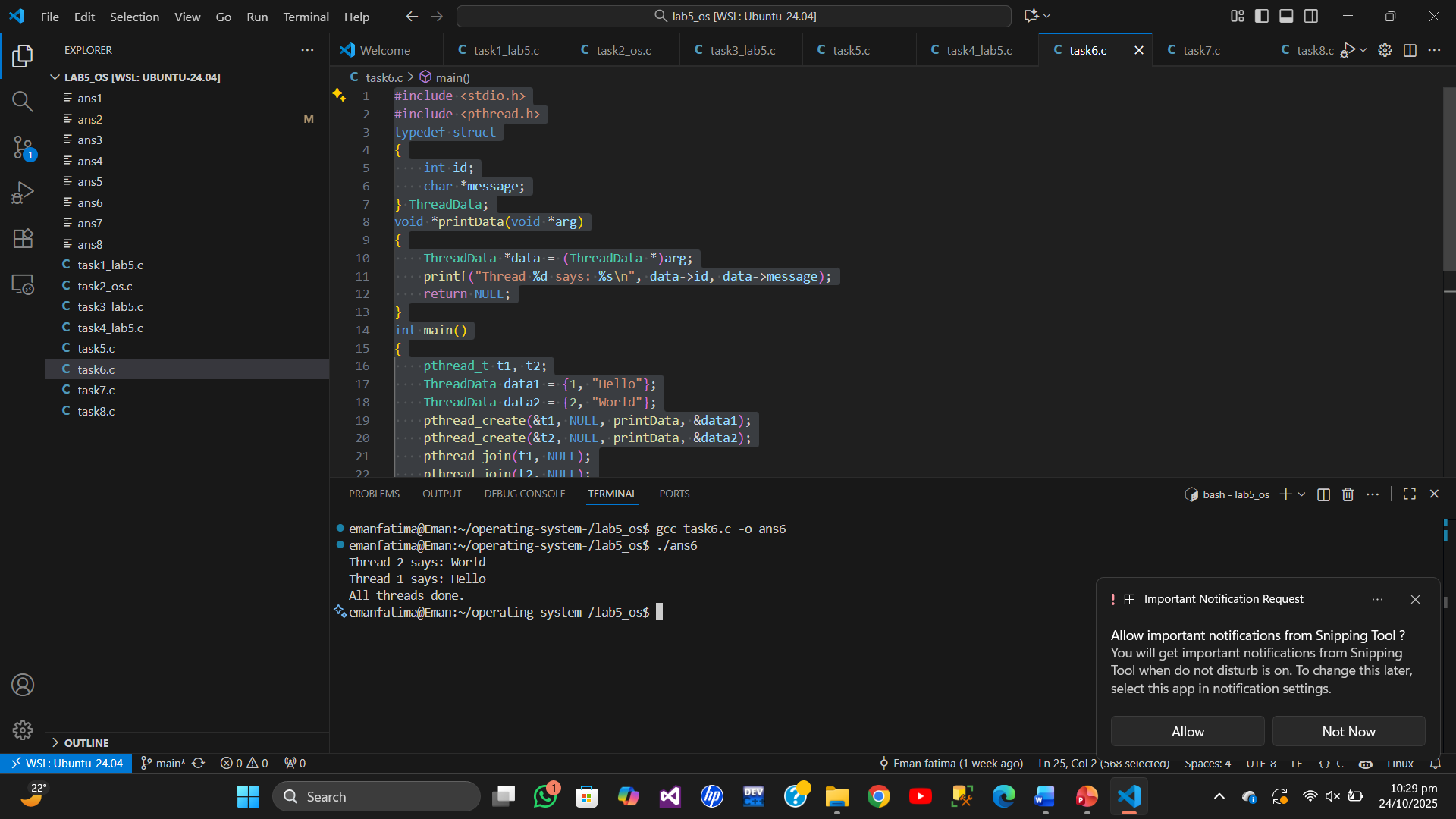
    pthread\_join(t1, NULL);

    pthread\_join(t2, NULL);

    printf("All threads done.\n");

    return 0;

}



Task 7:

#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;

}

A screenshot of a computer program

AI-generated content may be incorrect.

Task 8:

#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;

}

