**Thread Creation and Execution**

**LAB # 07**

****

**Spring 2021**

**CSE204L Operating System**

Submitted by: **Ashfaq Ahmad**

Registration No: **19PWCSE1795**

Class Section: **B**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Student Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Submitted to:

**Engr. Mian Ibad Ali Shah**

August 19, 2021

**Department of Computer Systems Engineering**

**University of Engineering and Technology, Peshawar**

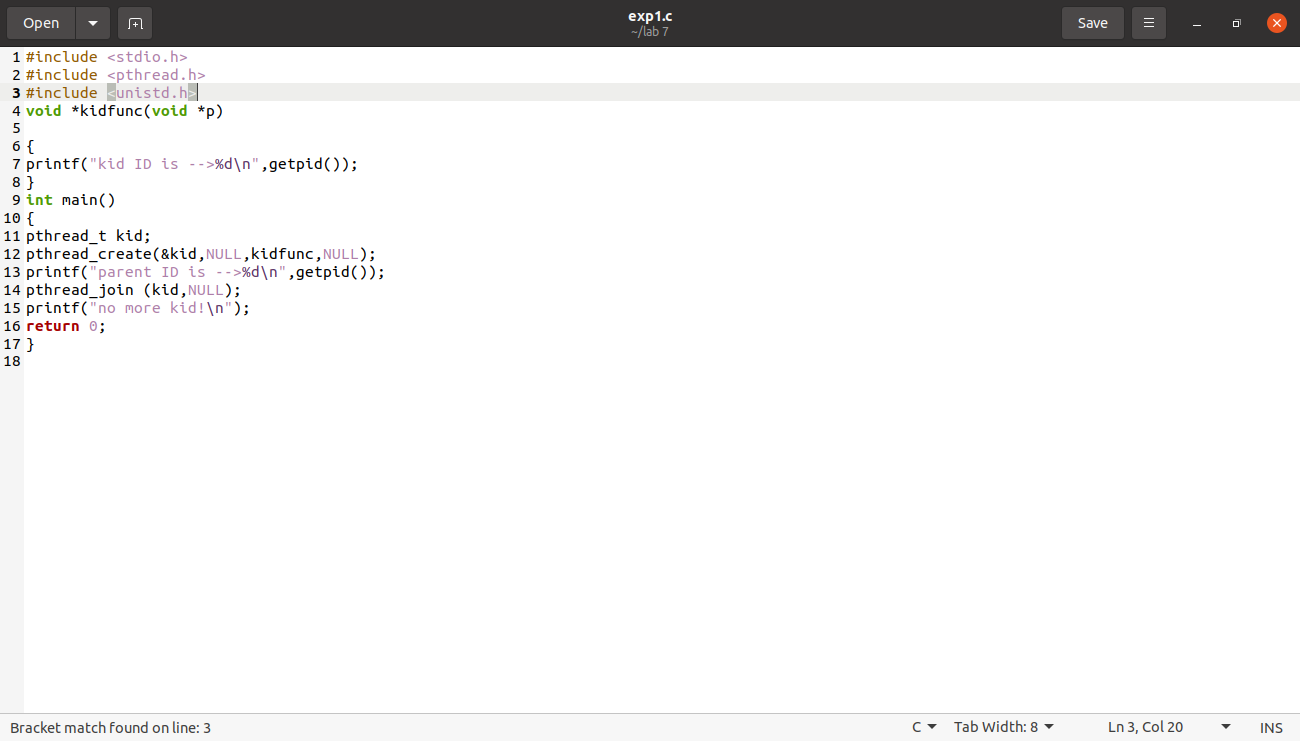
**Lab Objective(s):**

* This lab examines aspects of **threads** and **multiprocessing** (and **multithreading**).
* The primary objective of this lab is to implement the Thread Management.

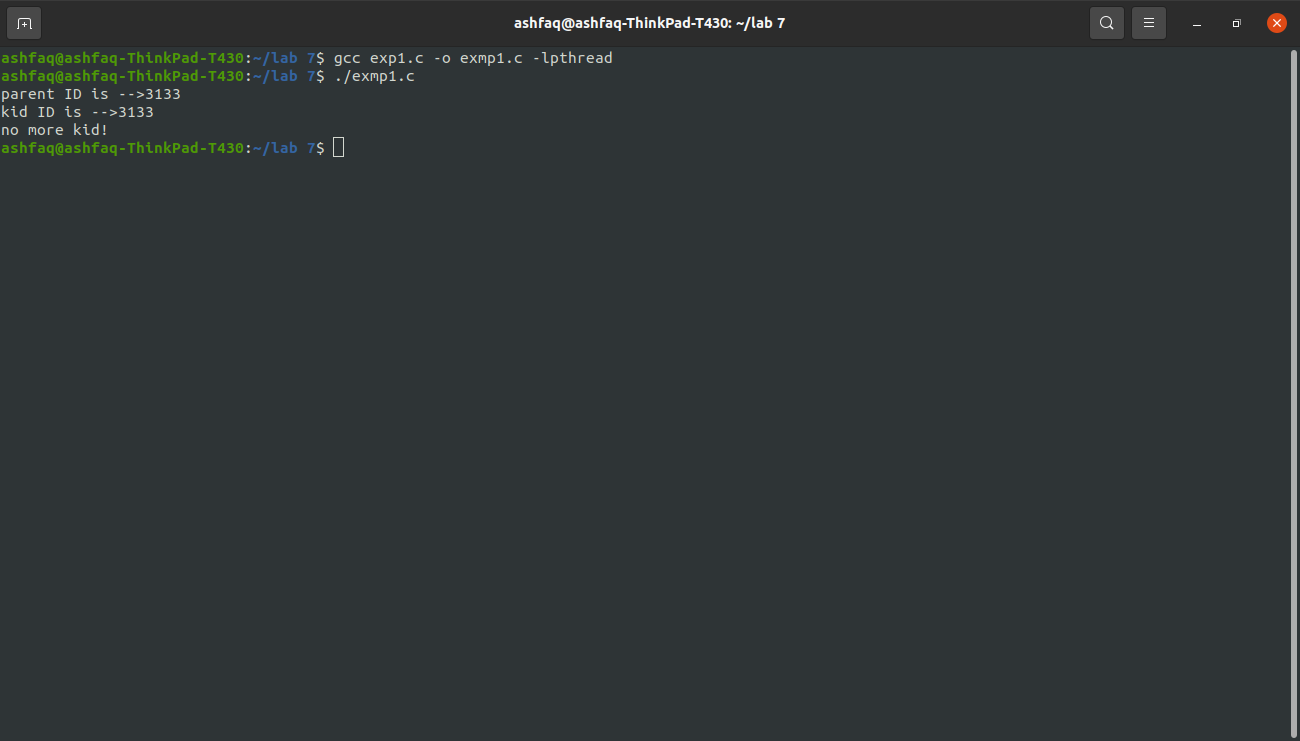
**Example # 01:**

**Are the process id numbers of parent and child thread the same or different? Give reason(s) for your answer.**

**Code:**



**Output:**



**Observations:**

Yes, the process id numbers for parent and child thread are the same because a thread is also a part of the process.

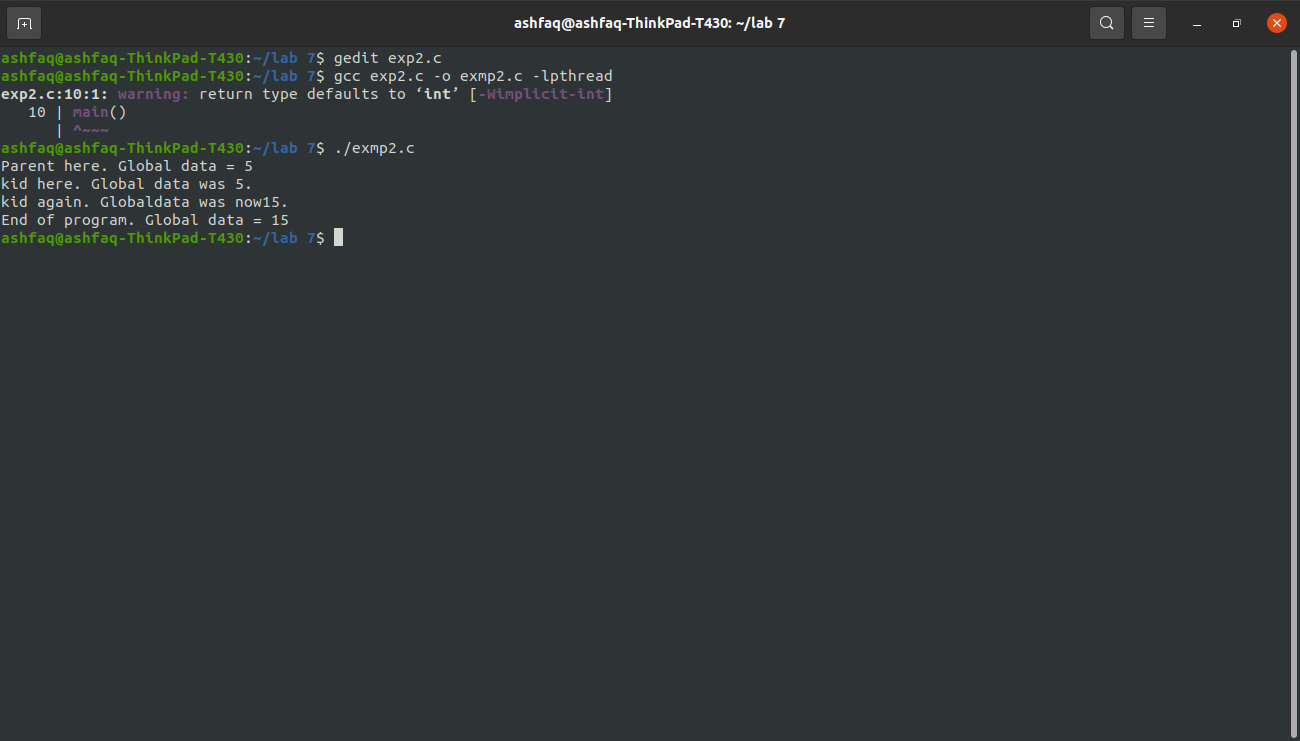
**Example # 02:**

**Do the threads have separate copies of glob\_data?** Why? Or why not?

**Code:**

****

**Output:**

****

**Observations:**

No, threads do not have different copies of global data because they share the same memory. Although each thread has its own stack but their global data is same as global data is stored in memory and not in stack.

**Example # 03:**

**Code:**

****

**Output:**

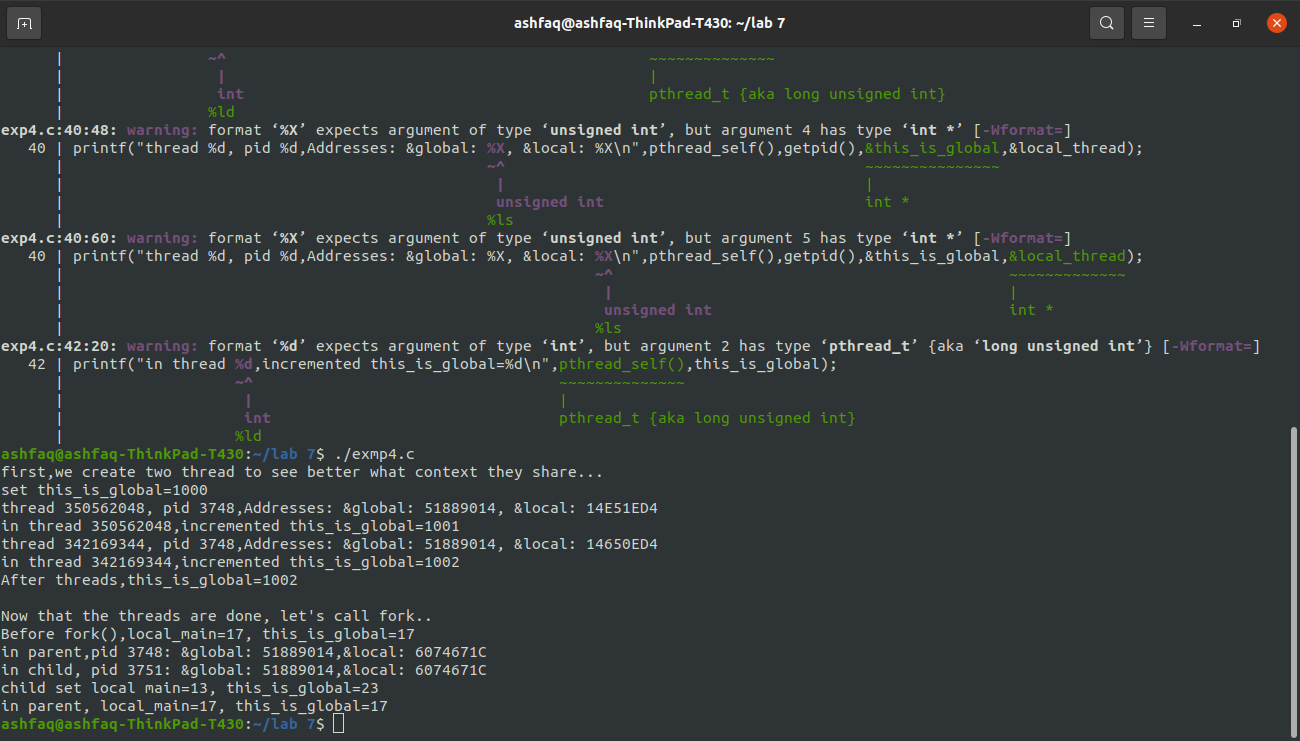


**Example # 04:**

**Code:**



**Output:**

****

**Task # 01:**

**Compile and execute the Box #1 program and show the output and**

**Explain why the output is so?**

**Code:**



**Output:**



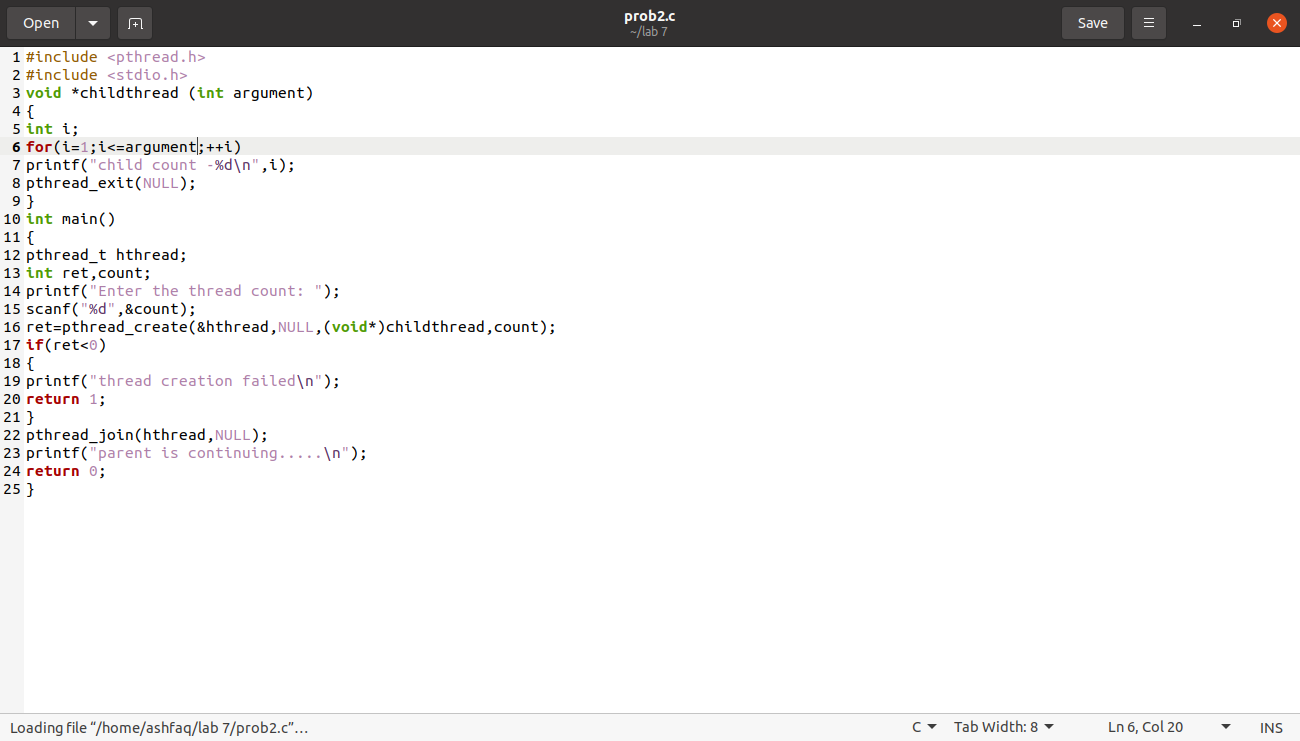
**Observations:**

Inside main a thread is created and the Child Thread function is passed to it . Due to the pthread\_join function the main thread waits for the child thread to terminate. A for loop is implemented inside the child thread which counts from 1 to 20.

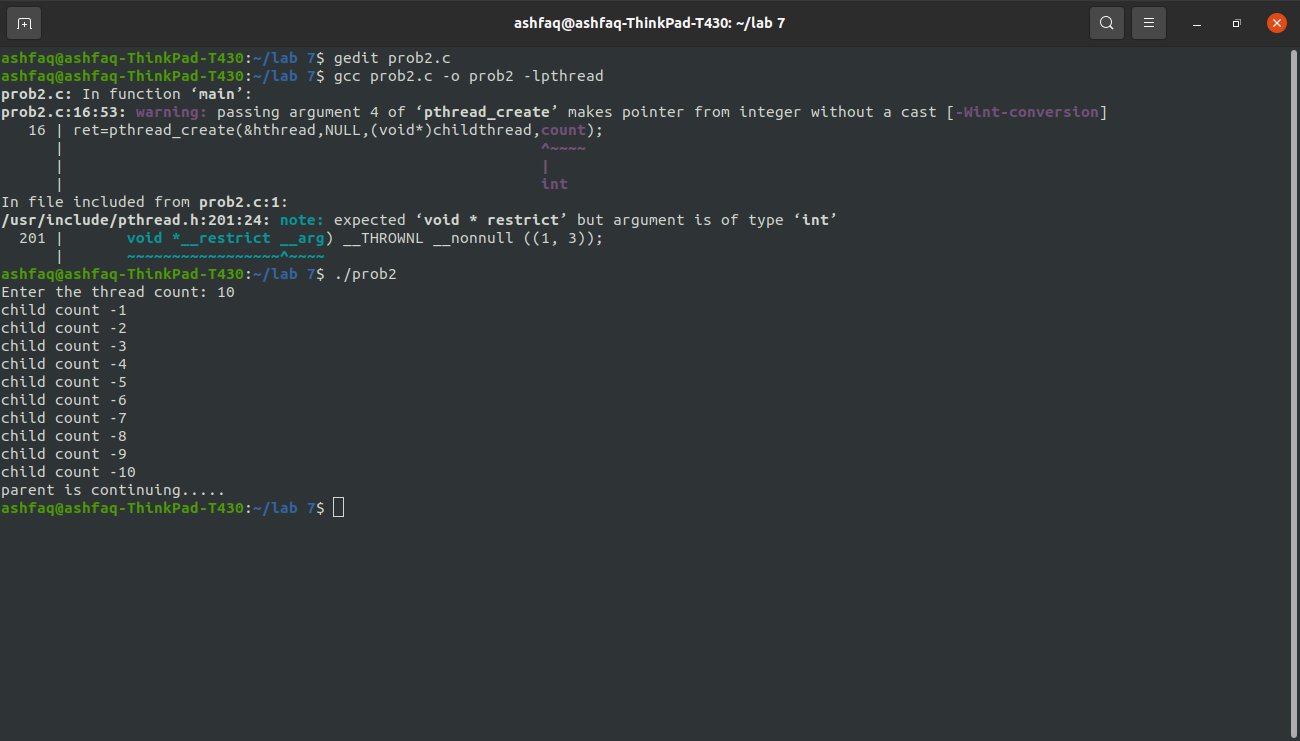
**Task # 02:**

In the **Box #2** modify the above **Box #1** program such that the main program passes the **count** as argument to the child thread function and the child thread function prints that many **count** print statements.

**Code:**



**Output:**



**Observations:**

In this program we are passing the count as an argument to the child thread. A for loop is implemented inside the child thread which counts from 1 to count.

**Task # 03:**

Write a program Box #3 by removing pthread\_exit function from child thread function and check the output? Is it the same as output of Box #2? If so Why? Explain?

**Code:**



**Output:**

******

**Observations:**

After removing pthread\_exit function from the given program the output is same as Task#2 because even if pthread\_exit function is not called explicitly in the child thread, when the child thread is done with its task it calls pthread\_exit implicitly.

THE END