CS 3305A: Operating Systems
Department of Computer Science
Western University
Assignment 3
Fall 2023
Due Date: October 24, 2023

Purpose

The goals of this assignment are the following:

- Get experience with *pthread* system functions.
- Learn how to create multiple threads for different tasks.
- Learn how different threads can access shared data.
- Gain more experience with the C programming language from an OS perspective.

Inter-Thread Communications (100 points)

Write a C program that will accept two integers from the user as **command-line arguments** (for example, X and Y where X, Y are positive integers). The parent process will read X and Y from the command line. The parent process will create three threads (i.e., thread_1, thread_2, and thread_3). The parent process will write X and Y to input_array[0] and input_array[1], respectively. The first thread (i.e., thread_1) will read X and Y from the input_array[] and perform the summation, S = X + Y, and then the result S will be written to the input_array[2]. Next, the second thread (i.e., thread_2) will read S from the input_array[2] and identify whether S is an even or odd number. Also, thread_2 will read X and Y from input_array[] and then perform multiplication M = X * Y and then write M to input_array[3]. Finally, the third thread (i.e., thread_3) will read M from the read_input_array[3] and reverse the number M. The expected output from your program should look like the following (for this example below, X and Y represent 21 and 3, respectively):

1. parent (PID 280448) receives X = 21 and Y = 3 from the user
2. parent (PID 280448) writes X = 21 and Y = 3 to input_array[]
3. thread_1 (TID 140451159217984) reads X = 21 and Y = 3 from input_array[]
4. thread_1 (TID 140451159217984) writes X + Y = 24 to the input_array[2]
5. thread_2 (TID 140451159217985) reads 24 from the input_array[2]
6. thread_2 (TID 140451159217985) identifies that 24 is an even number
7. thread_2 (TID 140451159217985) reads X and Y from input_array[],writes X * Y = 63 to input_array[3]
8. thread_3 (TID 140451159217986) reads 63 from input_array[3]
9. thread_3 (TID 140451159217986) reverses the number 63 → 36

In the above example, in line number 6, if S is NOT an even number, then the phrase "**identifies that S is an even number**" above should read as "**identifies that S is an odd number**". You **must**

control the execution of the threads to follow the sequence order shown in the sample output above. Your implementation must have the following functions:

1. void *sum(void *thread_id): This function is executed by thread_1. This function reads X and Y from input_array[], performs summation i.e., S = X+Y, and writes S to input_array[2].

2. void *even_odd(void *thread_id): This function is executed by thread_2. This function reads S from the input_array[2] and identifies whether S is an even or odd number.

3. void *multiplication(void *thread_id): This function is executed by thread_2. This function reads X and Y from input_array[], performs multiplication i.e., M = X*Y, and writes M to input_array[3].

4. void *reverse_num(void *thread_id): This function is executed by thread 3. This function reads M from input_array[3] and reverses the number M.

Mark Distribution

Inter-Thread Communications (100 points):

a) Parent reads X and Y from user: 10 points
b) The first thread reads X and Y from input_array[]: 10 points
c) The first thread adds X and Y and writes results S to input_array[]: 10 points
d) The second thread reads S from the input_array[]: 5 points
e) The second thread identifies whether S is an odd / even number: 15 points
f) The second thread reads X and Y, multiplies X and Y, and writes results M to input_array[]: 15 points
g) The third thread reads M from the pipe: 5 points
h) The third thread reverses number M: 15 points
i) Control the thread execution flow: 15 points

You must pass the input to the program using the command line argument. **The hardcoded input will not be accepted** and marks will be deducted accordingly.

Computing Platform for Assignments

You are responsible for ensuring that your program compiles and runs without error on the computing platform mentioned below. **Marks will be deducted** if your program fails to compile or runs into errors on the specified computing platform (see below).

- Students have virtual access to the MC 244 lab, which contains 30 Fedora 28 systems. Linux machines available to you are **linux01.gaul.csd.uwo.ca** through **linux30.gaul.csd.uwo.ca**.
- It is your responsibility to ensure that your code compiles and runs on the above systems. You can SSH into MC 244 machines.

- If you are Off-Campus, you have to SSH to **compute.gaul.csd.uwo.ca** first (this server is also known as sylvia.gaul.csd.uwo.ca, in honor of Dr. Sylvia Osborn), and then to one of the MC 244 systems **(linux01.gaul.csd.uwo.ca** through **linux30.gaul.csd.uwo.ca**).
- https://wiki.sci.uwo.ca/sts/computer-science/gaul

Assignment Submission

You need to submit only one C file. The name of your submitted C file must be "assignment3.c". Marks will be deducted if your submitted C file name is different. You must submit your assignment through OWL. Be sure to test your code on one of MC 244 systems (see "Computing Platform for Assignments" section above). **Marks will be deducted** if your program fails to compile or runs into errors on the computing platform mentioned above.

Assignment 3 FAQ will be made available on OWL as needed. Also, consult TAs and the Instructor for any questions you may have regarding this assignment.

Good Luck!!