LAB 11

CS 361: Systems Programming / Spring 2023

Description

In this lab session, you will explore how to avoid shared state between threads, and how to synchronize threads with semaphores and mutexes.

Guide

1. Accept the invitation for Lab 11 on Github classroom: https://classroom.github.com/a/hCQxuVr_
2. Import the Github repository created to your machine using vscode, as explained in Assignment 0
3. Make sure that you can launch a terminal inside vscode via menus: Terminal > New Terminal
4. Read this guide and answer the questions as they appear. You should answer a total of 10 questions.

Shared variables

Please open the slides for Lecture 19 about what variables are shared between many threads.

Question 1: Inspect file lab11-1.c and list all variables that can be shared between threads.

Question 2: Compile (using make) and run lab11-1 (using ./lab11-1). What does it print? Does the output change if you run this program many times (e.g., 10) ?

Question 3: Modify file lab11-1.c such that functions look_for_as and look_for_bs run in separate threads. Your file should launch both threads first, then join them and get the return (check Lab 10 if you don't remember how to do this). Now run the multi-threaded program. What does it print? Does the output change if you run this program many times (e.g., 10) ?

Question 4: Turn variable ret into a private variable, that is not shared between threads. Re-compile and , re-run the multi-threaded program. What does it print? Does the output change if you run this program many times (e.g., 10) ?

Question 5: Eliminate all variables that are shared between threads, by moving them somewhere else that is private to each thread. What did you change?

Question 6: After Q4, re-run the multi-threaded program. What does it print? Does the output change if you run this program many times (e.g., 10) ?

Mutexes

Please open the slides for Lecture 20 about mutexes and semaphores.

Question 7: Compile (using make) and run lab11-2 (using ./lab11-2). What does it print? Does the output change if you run this program many times (e.g., 10) ?

Question 8: Inspect the contents of lab11-2.c and change the loop that calls function thread to create threads instead. Add another loop to wait for all threads to finish. Now re-compile and run the multi-threaded program. What does it print? Does the output change if you run this program many times (e.g., 10)?

Question 9: Replace the body of the loop in function thread with a single increment (e.g., var++). Now re-compile and run the multi-threaded program. What does it print? Does the output change if you run this program many times (e.g., 10)?

Question 10: Add a mutex accessible to all code in file lab11-2.c, initialize it before starting all threads, and use lock/unlock around each variable increment. Now re-compile and run the multi-threaded program. What does it print? Does the output change if you run this program many times (e.g., 10)?

Extra / Optional

1. Why does the result of Q10 run slower than Q9?
2. How would you use a semaphore instead of a mutex to achieve the same result?

Grading

Show your UIC card to the TA when you enter the lab, or type your UIN on the chat when joining remotely. Stay in the session until you show your work, or until the TA announces that the lab is over.

- You have to remain present for the whole lab to get attendance, which you can then use to resubmit Assignment 5.
- You can leave early after showing your work to the TA (answers to all questions). In this case, you will get a 5% bonus in Assignment 5.