Winter 2022
CS3413
Lab 3

1. **For this lab, please use the skeleton C file provided on D2L. Part of the skeleton is commented out, as you are to find where the functions are supposed to be used.**
2. Write a C program to create two child processes using *pthread_create()*. Create a global *integer* variable (**gv** initialized to 1) that should be incremented by one 100 times by one pthread while the second pthread does the same but **decrements** the same variable by one. Have the main function wait for the two pthreads to complete and then print the value of the variable.
3. Run your program several times. Do you always see 1 as the result? Explain in the comments in the file.
4. Alter your program for the threads to repeat addition and decrementing 1000 times
5. Run your program several times. Do you always see 1 as the result? Explain in the comments in the file.
6. Alter your program for the threads to repeat addition and decrementing 10000 times
7. Run your program several times. Do you always see 1 as the result? Explain in the comments in the file.
8. Modify your program so that your child processes protect the global variable using *pthread_mutex_lock()* and *pthread_mutex_unlock()*.
    a. You can declare your mutex as a global variable
    b. Do not forget to initialize your mutex in the main before using it and destroy it after you no longer need it. From the manual:
        i. pthread_mutex_init(&mutex, NULL);
        ii. pthread_mutex_destroy(&mutex, NULL);
9. Run your program several times. Do you always see 1 as the result? Explain in the comments in the file.
10. Your task is to uncomment the plus and minus finctions, run the program and modify the plus and minus functions by using the semaphores *sem_wait()* and *sem_signal()* to make sure that + and – are always altering. Hint: Initialize the value of semaphore to 0; Join the two threads.
11. Run the program multiple times to make sure it's always the same.
12. For a moment, comment out the usleep() statement in the plus function, run the program multiple times, observe the output. Realize that sleep "helps" synchronization, as nothing prevents the plus function from signaling the semaphore over and over again. Note that relying on sleep for synchronization is bad, as it introduces delay that potentially can be avoided.
13. Uncomment the usleep statement. Reduce the maximum number of iterations of a loop in a plus function to three.

14. Compile and run your program, explain in the comments in the file what happens.

15. Submit your solution with comments to D2L