Winter 2020
CS3413
Lab1

Processes

1. Write a C program to create two child processes using *pthread_create()*. Create a global *int* variable (initialized to 0) that should be incremented by 1 (100 times) by one child process while the second child process does the same, but **decrements** the same variable. Have the parent of the children wait for the two children to complete and then print the value of the variable.
2. Run your program several times. Do you always see 0 as the result? Explain.
3. Trace system calls used by your program using `strace` *nameofyourexecutable* (-c option gives a summary of system calls only)
4. Identify the system call responsible for creation of your child process.
5. Modify your program so that your child processes protect the global variable using *pthread_mutex_lock()* and *pthread_mutex_unlock()*.
6. Run your program several times. Do you always see 0 as the result? Explain.
7. Trace system calls used by your program using `strace` *nameofyourexecutable* (-c option gives a summary of system calls only)
8. Identify the system call responsible for locking and unlocking the mutex.
9. Submit your final program using a mutex on D2L.

Don’t forget when writing pthread programs!

To include the pthread.h library:
#include <pthread.h>

To compile, add -lpthread to the linker flags:

```
gcc –lpthread –g  <my_filename.c>
```