Task 1

joer@joer-VirtualBox:~/os\_lab\_2019/lab5/src$ gcc mutex.c -o mutex.out -pthread

joer@joer-VirtualBox:~/os\_lab\_2019/lab5/src$ ./mutex.out >> out.txt

Take some interesting (ft - first thread, st - second thread):

counter = 3

doing one thing

counter = 4

doing another thing

counter = 3 //while ft had increment the work var, st had saved it.

doing another thing

counter = 4

And this are similar situation:

doing one thing

counter = 15

doing one thing

counter = 16

doing one thing

counter = 17

doing one thing

counter = 18

doing another thing

counter = 15

doing another thing

counter = 16

If we use mutex the common var is saved for thread multiuse. Data race end. In every iteration counter value incremened.

Task 2

joer@joer-VirtualBox:~/os\_lab\_2019/lab5/src$ gcc factorial.c -o factorial.out -pthread

joer@joer-VirtualBox:~/os\_lab\_2019/lab5/src$ ./factorial.out --k=10 --pnum=4 --mod=10

Factorial 10 mod 10 is equals 0.

joer@joer-VirtualBox:~/os\_lab\_2019/lab5/src$ ./factorial.out --k=4 --pnum=4 --mod=10

Factorial 4 mod 10 is equals 4.

joer@joer-VirtualBox:~/os\_lab\_2019/lab5/src$ ./factorial.out --k=7 --pnum=1 --mod=13

Factorial 7 mod 13 is equals 9.

Task 3

gcc deadlock.c -o deadlock.out -pthread; ./deadlock.out