Introduction to Operating Systems
METU Computer Engineering

Programming Assignment # 2

Instructor: Yusuf Sahillioğlu

Deadline: 23.05.2021 23:59

(20% of the actual grade)

Your code will be tested by Moss against cheating attempts, any cases suspected of plagiarism will result in total loss of grade and might result in further disciplinary actions.

Please submit your code as a C file on ODTUCLASS before the deadline.

You'll work on thread synchronization in Unix using POSIX thread (pthread) library.

Part 1 [50 points]: Synchronize $d \leq m$ threads so they can shift an m x m matrix concurrently. You will repeat the following shifts s many times: first perform a circular shift from left to right, then a circular shift from bottom to top. Here is an example with $m = 4$ and $s = 1$:

7	6	5	3
8	2	4	5
7	3	1	9
2	1	0	8

3	7	6	5
5	8	2	4
9	7	3	1
8	2	1	0

5	8	2	4
9	7	3	1
8	2	1	0
3	7	6	5

Each thread is responsible for m/d consecutive rows (last thread may take more if $d \nmid m$) during row shifting, and then m/d consecutive columns for the column shifting, e.g., with $m = 4$ and $d = 2$, thread 0 shifts rows 0-1, and thread 1 shifts rows 2-3. No thread can start shifting columns until all threads have finished shifting rows. Also, if $s > 1$, no thread can start shifting rows for the current stage until all threads have finished shifting columns in the previous stage.

You will read d and s from keyboard, and the input matrix from input.txt file which begins with m and then one row per line. Main thread prints the input matrix as well as the shifted matrix.

Part 2 [50 points]: Convert the monitor-based solution of the Dining Philosophers problem into a new solution without monitors using mutex and condition variables of pthread. As a novelty, require each philosopher to obtain 3 forks to start eating, 2 from left and right and 1 from a box of separate forks that initially contains $f = d / 3$ forks, where d is the number of philosopher threads read from keyboard. This means that at most one third of the philosophers can be eating at any given time. Let each philosopher sleep a random number of seconds (max 5) during their thinking and eating periods and also let them eat 50 times at most.

Describe your solution clearly in the beginning of your C file as a comment. Print the ID and state of each philosopher as soon as the state changes. Print a warning when there is only 1 fork left at the box. When all philosophers are done, print the average HUNGRY state duration over all philosophers, and ID of the philosopher with the largest HUNGRY state duration.