ISE 302–Operating Systems 15.11.2018

2018–2019, Homework 2

Introduction

Imagine that threads represent dancers and that two kinds of dancers, leaders and followers, wait in two queues before entering the dance floor. When a leader arrives, it checks to see if there is a follower waiting. If so, they can both proceed. Otherwise it waits. Similarly, when a follower arrives, it checks for a leader and either proceeds or waits, accordingly. Each leader can invoke dance concurrently with only one follower, and vice versa.

You will code the exclusive queue solution for the explained problem. The pseudo codes for leaders and followers are given below.

Exclusive Queue solution (leaders):

```
mutex.wait ()
if followers > 0:
    followers --
    followerQueue.signal ()
else :
    leaders ++
    mutex.signal ()
    leaderQueue.wait ()

dance ()
rendezvous.wait ()
mutex.signal ()
```

Exclusive Queue solution (followers):

```
mutex.wait ()
if leaders > 0:
    leaders --
    leaderQueue.signal ()
else :
    followers ++
    mutex.signal ()
    followerQueue.wait ()

dance ()
rendezvous.signal ()
```

The variables to be used in the solution:

```
leaders = 0
followers = 0
mutex = Semaphore (1)
leaderQueue = Semaphore (0)
followerQueue = Semaphore (0)
rendezvous = Semaphore (0)
```

Program Input

Program inputs are the number of leaders**(n)** and the number of followers**(m)**. These inputs should be received as a command line argument. To evaluate your programs, multiple options will be used.

./yourprogram n m

Example command: **./yourprogram 5 6**

Implementation Details

1. Your program should create n + m threads where n and m are provided by the user as command line arguments. n and m show the number of leaders and the number of followers, respectively.

2. Leader and Follower threads should print the following messages when appropriate:
   - Leader **i**: No available follower, so I wait. There are other **k** leaders waiting.
   - Leader **i**: **k** followers are waiting, so I signal the next follower in the queue.
   - Follower **i**: No available leader, so I wait. There are other **k** followers waiting.
   - Follower **i**: **k** leaders are waiting, so I signal the next leader in the queue.
   - Leader **i**: We are dancing together now.
   - Follower **i**: We are dancing together now.
   - Leader **i**: I leave now.
   - Follower **i**: I leave now.

3. "dance()" line in the pseudo code should be replaced with usleep(d) where d is 500000.

4. Also add a random delay (usleep) to the beginning of the thread functions. Delay duration should be between 0 and 2000000. It should be different for each thread and in each run.

Include development environment information, compilation and running commands as a comment in your code.

EXAMPLE:

```
// Development environment: Lubuntu 16.04 or ITU SSH servers
// To compile: g++ -c Homework2.cpp –pthread
// To run: ./a.out n m
// Example: ./a.out 5 6
```

Appendix

You may make use of the following code examples for reading command-line arguments and converting strings to numbers in C and C++.

- https://rosettacode.org/wiki/Command-line_arguments
- https://www.geeksforgeeks.org/converting-strings-numbers-cc/

Academic dishonesty including but not limited to cheating, plagiarism, collaboration is unacceptable and subject to disciplinary actions.