Operating Systems **CSI3131** ***Assignment 2*** *Winter 2020*

Semaphores/inter-thread synchronization

You must submit your assignment on-line with Virtual Campus. This is the only method by which we accept assignment submissions. Do not send any assignment by email. We will not accept them. We are not able to enter a mark if the assignment is not submitted on WebCT! The deadline date is firm since you cannot submit an assignment passed the deadline. You are responsible for the proper submission of your assignments and you cannot appeal for having failed to do so. A mark of 0 will be assigned to any missing assignment.

Assignments must be done individually. Any team work, and any work copied from a source external to the student (including solutions of past year assignments) will be considered as an academic fraud and will be forwarded to the Faculty of Engineering for imposition of sanctions. Hence, if you are judged to be guilty of an academic fraud, you should expect, at the very least, to obtain an F for this course. Note that we will use sophisticated software to compare your assignments (with other student's and with other sources...). This implies that you must take all the appropriate measures to make sure that others cannot copy your assignment (hence, do not leave your workstation unattended).

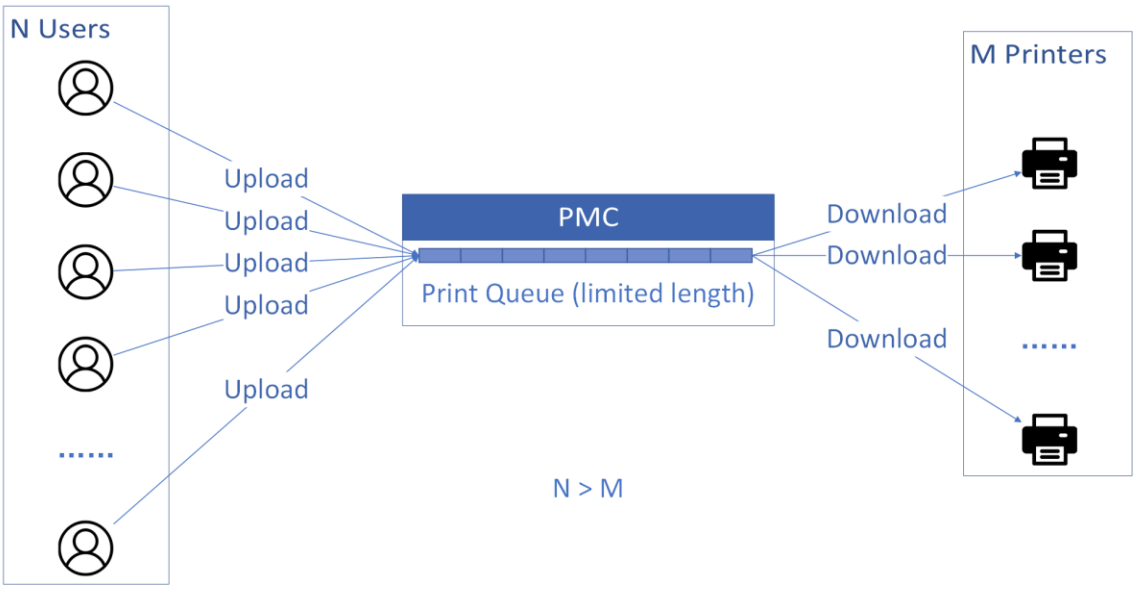
Goal: Practise semaphore usage.

Posted:
Due:

Description (Please read the complete assignment document before starting.)

Simulating a Print Management Center

Consider a Print Management Center (PMC) used for managing printers. As shown in fig.1, a library has several printers sharing the same print queue maintained by a PMC. When users upload files to PMC, PMC will add these files to print queue. Meanwhile, all the available printers will try to download files from PMC. Once the queue is not empty, PMC will poll files from the print queue.



User:
MaxLoop
Loop for MaxLoop times {
Create UserFile
Upload UserFile
}

Printer:
for(;;) {
Download UserFile
Print UserFile
}

PMC:
Upload(UserFile):
Acquire Semaphore
Acquire Mutex
Add UserFile to Print Queue

Release Mutex

Download:

Acquire Mutex

Poll UserFile from Print Queue

Release Mutex

If UserFile != null:

Release Semaphore

Return UserFile

Marking Criteria:

The following conditions must be satisfied in order to get marks:

- 1. Initialize Users and Printers +10.
- 2. Start Users and Prints Threads +10.
- 3. Shutdown Users and Printers +10.
- 4. User Loop +10.
- 5. Create Semaphore, Mutex, and Queue +10.
- 6. Upload Function +20.
- 7. Download Function +20.
- 8. Printer Loop +10

Your goal:

Take the skeleton code and complete the missing part. Your goal is to provide proper synchronization between users and printers, satisfying the conditions for correct simulation. Use only semaphores to achieve the needed synchronization.

Hints:

- Take a look here for many semaphore problems and solutions.
- On Unix/Linux systems (and on some Window systems as well), the following commands might help you with debugging/verifying proper execution:

    1. Launch the program with the output redirected to a log file, i.e. “java Assignment2 > assignment2.log”

2. To see what user 5 did, type “cat assignment2.log | grep “User 5”

3. To see what printer 3 did, type “cat assignment2.log | grep “printer 3”

4. This way, you can rather easily see if your solution does not work yet.

5. You can figure out for yourself other ways to filter out useful information from the log file, or simply study the log file to see whether everything worked as specified

To submit your solution:

Submit single file assignment2.java, containing your solution. You will get partial marks for partial solutions. Be sure to include your name and student number at the start of the file.