**Operating system 2 Project – Cover sheet**

Project Title: Reader- writer project

Group: #-----------------

Discussion time: 12:20 PM

Instructor : ENG:abdelrahman

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | NAME | BOUNCE | MINUS | TOTAL GRADE | COMMENT |
| 202000160 | Amir Samir fawzy(T.L) |  |  |  |  |
| 201900882 | Mina mofed mouris |  |  |  |  |
| 201900433 | Abdlelrahman mohmed |  |  |  |  |
| 202000621 | Fady osama akram |  |  |  |  |
| 201900396 | Abdleltawab ahmed |  |  |  |  |
| 201900873 | Mirna monir moussa |  |  |  |  |
| 201900417 | عبدالرحمن خليفة حلمي خليفة |  |  |  |  |
| 20180014 | ا براهيم مجدى ابراهيم حسن |  |  |  |  |

**Solution pseudocode:**

Step(1): Reader get lock

Step(2): increment reader count of readers by 1

Step(3) : check if reader count equal 1

Step(4) :Writer acquire lock

Step(5) : Reader release the lock

Step(6) : Reader Enter the critical section

Step(7): Reader acquire lock

Step(8): decrement reader count of readers by 1

Step(9) : check if reader count equal 0

Step(10) : Writer acquire lock

Step(11) : Reader release the lock

Step(12) :Writer acquire the lock

Step(13) : Writer Enter the critical section

Step(14) : Writer release the lock

**Examples of Deadlock :**

**1) Ordering locks**

• occur when two threads attempt to acquire the same locks in a different order

• A program will be free of lock-ordering deadlocks if all threads acquire the locks they need in a fixed global order

• A program that never acquires more than one lock at a +me will also never deadlock

Text

Description automatically generated with low confidence

**The solution :**

The solution we have used is semaphore….all reader threads will enter the critical section in order so every thread will enter the critical section in its order so no thread will enter in another thread order

****

**(2Examples of starvation :**

what if there are many writers who arrive before any readers and the first writer took a very long time to finish his writing then r maybe reach some big minus number let’s say -12345 and after it, readers begin to arrive along with writers, and somehow the operating system always choose a writer to take the semaphore and not reader, in that case, the readers would be starving

**Graphical user interface

Description automatically generated with low confidence**

**The solution :**

Note that this solution can only satisfy the condition that "no thread shall be allowed to starve" if and only if semaphores preserve first-in first-out ordering when blocking and releasing threads. Otherwise, a blocked writer, for example, may remain blocked indefinitely with a cycle of other writers decrementing the semaphore before it can.

****

**Real-world examples :**

Banking system : read account balances versus update :

The bank has several cash machines, all of which can read and write the same account objects in memory.

Of course, without any coordination between concurrent reads and writes to the account balances, things went horribly wrong

To solve this problem with locks, we can add a lock that protects each bank account. Now, before they can access or update an account balance, cash machines must first acquire the lock on that account

**A picture containing diagram

Description automatically generated**

To solve this problem with locks, we can add a lock that protects each bank account. Now, before they can access or update an account balance, cash machines must first acquire the lock on that account.

In the diagram to the right, both A and B are trying to access account 1. Suppose B acquires the lock first. Then A must wait to read and write the balance until B finishes and releases the lock. This ensures that A and B are synchronized, but another cash machine C is able to run independently on a different account (because that account is protected by a different lock) 