MdShahid Emdad

12-20-2021

**Lab 5 – Report**

**Task:**

Identify the critical section and synchronize the program. You would need to use P(sem) and V(sem) operations at the right places to solve the synchronization problem. You will also need to find the logical flaw if there is any and try to solve it. To use P(sem) and V(sem) include the provided “sem.h” file. Your solution should solve the following problems:

* Prevent race conditions
* Prevent a son from withdrawing money when there is no balance. • Prevent undefined outputs like negative balance
* Prevent a process from continuously requesting access to the shared memory. Prevent unnecessary cpu cycle.
* The problem should be solved with as few semaphore variables as possible

Submit a report showing the critical section of the code ( and logical errors) and explain your solution in detail.

**Solution:**

I solved the critical sections by making the dad process, first son process, and second son process. Then, I prevented a son process from withdrawing money when there isn’t enough balance by adding a condition within their critical portion which also prevented undefined outputs from happening. Finally, I used only one semaphore. Please check my bank.c file to for the code.

In order to achieve this, In the sem.h file it is stated that the P operator has to be called upon entry of a critical region, and the V operator has to be called upon exit of a critical region. I followed the same setup in syncExample.c file. The first son process ends and I call the V operator. To ensure that the first son process does not withdraw money if there isn’t enough to withdraw, I added an if check that will check if the current balance is more than the withdrawAmount. The second son process ends and I call the V operator. To ensure that the second son process does not withdraw money if there isn’t enough to withdraw, I added an if check just like in the first son process.