Zeal Patel

CSC 33200 – S

Nov 12, 2020

**Lab 5 Report**

To prevent the race condition in the problem I implemented semaphores to synchronize the three processes. Before, the dad or either of the sons could go to the “bank” at the same time to deposit or withdraw money. After implementing semaphores using the sem.h library only one process can enter the “bank” or the critical section at once. Before entering the critical section I call the P() method from sem.h that decrements the semaphore value to -1 . This locks the critical section, so while the process that called P() is in the critical section the other two processes cannot enter it and alter the data. Once, the process is done with it’s job it calls the V() method from sem.h to increment the semaphore value to 1. This signals the two processes that the critical section is available for use.

|  |  |  |  |
| --- | --- | --- | --- |
| N | T(dad) | T(son1) | T(son2) |
| 5 | 297 | 125 | 189 |
| 10 | 3630 | 1751 | 1891 |
| 15 | 11566 | 430 | 736 |
| 20 | 14349 | 4045 | 10319 |
| 25 | 4816 | 501 | 4317 |