Tahsin Alam

Task 5

Task 5 Process Synchronization Dad-Son problem

There is a way to prevent race condition in the Dad-son problem with the use of a single semaphore (sem) and several P(sem) and V(sem) statements. There are three critical sections (CS). The first critical section is the dad depositing money into a balance. The other two critical section are both sons withdrawing money from the same balance that the dad is depositing. Race conditioning was prevented by enclosing the critical sections with the statements P(sem) before CS and V(sem) after CS. This will prevent other processes entering the CS when a process is already inside CS. The figures below shows that each process isn’t interrupted while it is executing.

To keep track of how many times each process has waited there needs to be three variables that act like counters. Each counter represents a process. The dad wait counter is waitTimeDad. The sons wait counter is waitTimeSon1 and waitTimeSon2. Once a process enters the cs, it increments the counters of the other two processes. An example would be the Dad is in CS, both sons will be waiting and their counters will be incremented. The data of the each counter is saved in a separate file since fork() will only create copies of the parent variables. Every time a process enters the CS it will access the counter files of the other two processes and increase their values by 1. The figure below shows how many times the Dad and sons have waited (the yellow highlighted rectangles).



