Jeffrey C. Slocum

Operating Systems Project #2

Matching Dice

In this project we used pthreads to synchronize the threads created to simulate the players and the dealer of this game. The game was not over until either Players A & C, or Players B & D rolled the same sum between 2 dice. The results were printed to a logfile, and rolls are printed to the console screen directly.

There were many challenges in this project. The main challenge being synchronization, and keeping the threads consistent with rolling the dice, and checking sums between the players before rolling the dice again. This was solved using pthread\_cond\_wait() and pthread\_cond\_signal(). A thread would decrement the semaphore, before proceeding to the critical section, and vice versa. In the critical section, the thread would increment the semaphore, allowing the other threads to proceed.

The next challenge faced was how to keep track of which players turn it was. This was needed in order for the dealer to know when to check for a winner, and then begin the next round of dice rolling. I used an array to keep track of the players information, which I passed as an argument to the thread function upon creation. This, id =\*((int \*) arg, enabled me to give each thread an id and allowed me to control the ‘flow’ of the game. I used a turn variable that combined with the thread id, allowed me to navigate which thread was to go next in rolling the dice. Once all players had rolled, the dealer was signaled to check for a winner of the game. If no winner, the player threads were signaled to begin rolling the dice again. This process will continue until 2 players have matching sums.

A picture containing object, clock, monitor, sitting

Description automatically generated

A screen shot of a computer

Description automatically generated

Graphical user interface, text

Description automatically generated

A screen shot of a computer

Description automatically generated

Graphical user interface

Description automatically generated