PROBLEM SET 2

MIGUEL VILLANUEVA

POSSIBLE DEADLOCK AND STARVATION EXPLANATION & SYNCHRONIZATION MECHANISMS USED TO SOLVE THE PROBLEM

Deadlock happens when multiple processes hold resources while waiting for the rest to release them. Starvation on the other hand happens when processes never get access to resources due to others utilizing all of it. I utilized semaphore to control access of the instances, this ensures that only n instances can be active at a time. Since the available parties acquires a semaphore before execution and releases after finishing, there is no circular waiting condition. The semaphore also allows for controlled execution, preventing some parties from being blocked. Another thing utilized is ThreadPoolExecutor that makes sure that the threads don't hold the resources indefinitely. The program utilizes a first-come, first-served scheduling algorithm. I also synchronized which prevent race conditions when updating instance status and it protects shared resources. There is also the synchronized (lock) which protects from concurrent modifications. There is also the threadpool that manages the execution of the parties and makes sure that the threads are reused.