race condition -> lock Mutex -> Mutual Exclusion Only I thread can have the self_lock = threading. Lock() with self. - lock: logging dead lock & RLock I

Lock bundle code up, during &

primitives execution, not swifth thread producer - consumer threating Threading & process synchronization Dusing 2 locks, first in first of pipeline only out stores I value () threading. Event queue. Queue semaphore 3 Better queue Timer Barrier

threading Jim 3/21/21 daemon will shut down immediately when the program exits. () x, 30in() wait thread to finish # comment out, still wait forit regular thread 1 daemon thread x.join() regular thread wait no matter x.join(), both daemon thread x.join() wait no join(), main thread does { main thread not wait start with threading. Thread A group of threads: executor, map . Submit Thread Pool Executor (max_workers=2)

randint (a,b) as reb

Producer - consumer problem

probably more difficult than the

concepts

Pipeline producer Consumer thread thread LOOP loop setmessage getmessage 2 locks LOOP ! loop Start before consumer_lock before start Set message producer_lock It has 2 locks has consumer-1 After ock After Release consumer_lock release produce 100P 2 Until producer_lock is realeased, wait it has producer lock