This is Steven Lambrou writing this README and I wrote the code for part2 in C. My design was simple and created a Queue for handling requests. Each node in the queue contained a request and a pointer to the next Node. I implemented an add and poll method for adding request nodes to the queue and removing nodes respectively. I also implemented a MasterInstructions struct to be passed to the master thread containing instructions such as maximum job number, maximum job length, and time to wait between jobs. These values are parameters set by the user running the program. The slave threads simply take in their ids as parameters to print out later. Two locks are used in the program. One to lock the queue and one to lock the CURRENT_TIME variable. The current time is only increased by the master thread when a job is created. Slave threads can only read and store the current time when they are assigned a job. For the queue locks the master thread locks the queue when adding things to it and releases once it has finished adding the request. The slaves wait for a job to be available in the queue by locking then running poll(). Poll() returns null when the queue is empty. Then releasing the lock on the queue after checking it. Once a slave thread has a request it prints the time it was received and then sleeps for the request duration. Upon completion the slave prints its id and the time the job finished. Sample output has been included in the solution.txt file.
Note the code uses the actual sleep method for this so if you type in large numbers for the joblength and job wait time, the code will take a long time to run for this reason. I wasn't sure if this was supposed to be fully simulated or use actual seconds so I chose actual seconds.