For both part A or part B, neither program ended up hitting a deadlock/livelock. For part A, the TAs (expectedly) kept messing with each other's work; the rubric kept changing and multiple TAs kept modifying the same line at the same time. Despite this, the processes still progressed and none of them got stuck in a deadlock/livelock. Eventually, student 9999's exam file was reached, where the TAs all exited.
For part B, the semaphore implementation made the TAs take turns accessing the shared memory. Every TA was able to update when needed one at a time. Due to the random delays, the order the TAs went changed from each time the process was run, but they all still made progress, eventually reaching student 9999 and exiting.