

## Part 2: C

Students: Matthew Gibeault 101323772 and Declan Koster 101301649

### Setup:

Part 2 a and c were both ran with 3 Tas and 20 exam files, with student numbers between 0001 and 9999. The logs are stored in output\_a.txt and output\_b.txt

### observations for part 2 A:

Rubric and first exam was loaded into shared memory at startup

[COORD] BEFORE READ exam file 'exam1.txt' (index=0)

[COORD] AFTER READ loaded exam for student 0001 (index=0, gen=1)

Tas review rubric, sometimes modifying and saving

[TA 1] BEFORE WRITE rubric[1] change E -> F

[TA 1] AFTER WRITE rubric[1] saved

Tas mark questions and print the student number

[TA 3] Marked question 1 for student 0001

exams advance sequestially with before and after logs

[TA 1] BEFORE loading next exam (index=0)

[COORD] AFTER READ loaded exam for student 1019 (index=1, gen=2)

program terminated when exam with student number 9999 is loaded

[COORD] AFTER READ loaded exam for student 9999 (index=19, gen=20)

[COORD] Sentinel exam (9999) detected. Signaling termination.

throughout the output no deadlock or livelock can be observed. All exams and processes exited

### Observations for part 2 B:

same sequence as part 2 A but this time with semaphores controlling critical sections

[TA 2] BEFORE WRITE rubric[4] change Q -> R|

[TA 2] AFTER WRITE rubric[4] saved

question marking guareded by sem\_question

[TA 2] BEFORE READ marked[0] = 0

[TA 2] AFTER WRITE marked[0] = 1, remaining=4

exam switching coordinated by sem\_exam only after all questions marked and program termination

when exam with student number 9999 marked

[COORD] Sentinel exam (9999) detected. Signaling termination.

why didn't deadlock nor livelock occur:

deadlock could occur if a TA held `sem_question` and then blocked on `sem_rubric` while another TA held `sem_rubric` and waited for `sem_question`. The program avoids this by never nesting locks and keeping each critical section minimal.

livelock could occur if TAs endlessly modify rubric without marking or spin forever waiting for exam changes. Logs show steady progress and short sleeps in spin loops, so no live lock occurred.

Execution order for each exam:

1. Coordinator loads exam into shared memory (BEFORE/AFTER logs).
2. All TAs review rubric, possibly apply single-character corrections, and save rubric.
3. TAs mark questions (1–5) with random delays and print student number.
4. When all questions are marked, one TA loads the next exam; others wait for `exam_generation` change.
5. When student 9999 is loaded, `terminate` is set and all TAs exit.