Part 2.c — Deadlock / Livelock Report:

Student Names: Uday Arya, Qurb E Muhammad Syed

Student Numbers: 101268848, 101281787

After running the semaphore-based implementation from Part 2(b) multiple times with different numbers of TA processes, we observed the following:

Deadlock:

No deadlock occurred in our run.

The reason is that each TA acquires only one semaphore at a time (for rubric writing, question marking, or exam loading), so the processes never create a wait condition resembling a deadlock. Because a circular wait is required for deadlock, the system did not enter a deadlocked state.

Livelock:

No livelock appeared during testing.
Although multiple TAs occasionally try to access the same semaphore, each eventually moves on due to randomized delays (0.5–1.0s for any rubric decisions and 1–2s for marking). This prevents processes from repeatedly retrying at the same time or endlessly reacting to each other.

Execution Order:

The processes interleave due to scheduler behavior and random delays. A typical pattern is:

All TAs will read the rubric concurrently.

Only 1 TA at a time updates the rubric and writes to the rubric file.

TAs then take turns marking any unmarked questions, coordinated by the marking semaphore.

When all questions of an exam are finally marked, a TA loads the next exam using the load semaphore.

Execution ends once an exam with student number 9999 is loaded.

Conclusion:

Across all test runs, no deadlock or livelock occurred.
The semaphore structure and randomized delays ensured that all TAs eventually made progress and the system terminated normally.