SYSC 4001 A3: Part 2 Report

Student 1: Hariharan Thennarasu, 101304027
Student 2: Raghav Ramaswamy, 101310114

- When the programs were run, all TA processes reached the desired endpoint and terminated after seeing an exam containing student number 9999. Since, no TA process was blocked permanently, no deadlock or livelock was observed in the programs.

Part A Observations)

In this part, because semaphores were not used to synchronize the processes, race conditions were in effect and caused messy output and undesired process behaviour. We will list and explain some of these problems below.

- **Question 5 not being marked**. One of the first problems experienced was that the TAs were not marking question 5. They would mark the first four and then prematurely load the next exam. This caused the TA who was supposed to mark question 5 of the past exam to mark question 5 of the new exam. Due to all TAs checking the exam concurrently, when a TA first claims question 5, the other TAs see this as question 5 being marked and load the next exam. This was fixed by adding a loader condition variable that only allowed one TA to load an exam at a time. Also, the loader condition could only become active after the current exam had been fully marked.

- **TAs load multiple exams and overwrite exam contents**. Due to a lack of synchronization, TAs were loading multiple exams at the same time, increasing the exam number each time. For example, shortly after exam2 was loaded into shared memory, another TA loaded the exam3 into memory. The more recently loaded exam overwrote the contents of the past loaded exam. This caused several exams to be skipped. The loader variable was used to fix this problem.

- **Not all TAs reviewed the rubric for each exam**. For the first two exams, the output was printed as desired, but afterwards, for certain exams, only some of the TAs would read and modify the rubric before marking the exam. This led to one TA marking exam questions while another was still modifying the rubric. This split up the marking questions output and went against the assignment specifications. This was fixed by introducing a review variable that ensured that each TA finished reviewing and modifying the rubric before marking any exam questions.

These were the major race condition problems that were encountered. We were aware that some race conditions were allowed to be present, but we decided to implement condition variables to manage the critical space. This is why the part A output looks clean and faces minimal race conditions.

Part B Observations)

In part A, we managed to keep the race conditions in check through specific variables like loader and review. Since part B allows us to use semaphores to enforce synchronization, we replaced the previously used condition variables with semaphores. By using semaphores, we ensured that only one TA can modify the rubric, load an exam, or mark a question at a time. Livelock was also avoided by always unlocking the semaphores after use, allowing waiting TAs to access and modify shared memory. While our condition variables allowed us to manage the critical section in part A, using semaphores allowed us to safely and correctly produce the required concurrency, output, and eliminate most race conditions.