Part 2.c)

- When running the program we decided to create 2 processes (2 TA's), you can also have more TA's if you like. The execution order is always progressing and constantly switches between both TA's. The parent first loads exam01.txt (student 1) into the shared memory, then both TA 1 and TA 2 start claiming, checking, and marking different questions of the same student exam. For example, TA 1 does questions (1 and 3) and TA 2 does questions (2, 4, and 5). Once one TA finishes its current question, it may either claim the next question or if all 5 questions are done, load the next exam file into the shared memory. This pattern repeats for all students: exams are processed in order from exam01 to exam21, but within each exam the questions are marked in random order by the two TA's depending on their random marking delays. At the end, TA 1 loads the last exam for student 9999, encounters the STOP marker and exits and TA 2 finishes its last question and then observes the STOP condition and exits. All TA's terminate normally with no deadlock and the shared memory is removed.