1. If we break down the project into functions or files, i came up with the following:
   1. Function to open, read and decode the list of read/write/ifetch given by Faust.
   2. create the logic/code behind the cycle count, relating the DRAM command cycles and CPU cycles.
   3. find the timing requirement for every DRAM command based on the current state.
      * this will be critical for code correctness and the testing part later as well
        + For example: Timing for a read after a read to the same bank. A write after read to a
   4. function that issues commands based on buffer state
   5. function to write to the trace file
2. After completing the code part, we move onto to testing.

* One of us comes up with the test cases, and the other tests them on the code.
* Cover all corner cases.

1. After our simulator is validated to our best ability, move on to the next level in the project.

If you would like to add to my understanding of the project, do so. It's easier to change things as we go instead of doing so later. And please be critical to my understanding and give me some constructive criticism if you can.

And this is how I would like to move forward.

Would you take step 1. part c) . Basically coming up with all the correct combinations for all possible

commands and their timing. In writing first and then we can discuss them. once we agree on that, we can code for it.

I'll focus on 1.a), 1.b).

My goal is to do noticeable progress daily and finish on time.

One last question, have you done OOP? i was thinking of coding the project in Ruby or Java. Let me know what you think:)