Exercise 1.1: multi-valued regular register

Consider the implementation of an $M$-valued one-writer $N$-reader (1WNR) regular register (Transformation III in the slides).

1. In the code of $write(v)$, is it possible to change the order of operations: first write 0 to $R[v-1], \ldots, R[1]$ and then write 1 to $R[v]$?

2. What if the writer writes 0 to $R[1], \ldots, R[v-1]$ in the ascending order? Justify your answers (e.g., by presenting an execution that violates the properties of a regular register).

3. If we replace the regular binary registers with atomic ones, would we get an implementation of an atomic multi-valued register? Justify your answer.

Exercise 1.2: atomic registers

1. Prove that the implementation of a one-writer one-reader (1W1R) atomic register is correct (Transformation IV in the slides).

   *Hint:* argue that to prove that the implementation is indeed linearizable, it is enough to show that if $read_1$ precedes $read_2$, then $read_2$ cannot return the value written before the value returned by $read_1$. Check the claim and the rest is trivial.

2. Consider the implementation of a one-writer $N$-reader (1WNR) atomic register (Transformation V in the slides).

   The code of $read()$ involves writing the value just read back to $RR[\_]$. Is it possible to devise an implementation in which the reader does not write?

3. Give a multi-writer multi-reader (NWNR) atomic register implementation from 1W1R atomic registers and sketch the proof of its correctness.

Exercise 1.3: Linearizability

*Bonus question:* An alternative definition of a completion of a history $H$ is any complete extension of $H$.

Is the definition of linearizability based on this notion of a completion equivalent to our original definition? Justify your answer.