**********
Homework
**********

1. (easy) Derive the version of lock framed wrt. the heap $\sigma_s$.

   \[\text{lock } l : [h]. \{ \sigma_s = h \land \alpha_s = \bot \land \mu_s = \text{own} \}
   \{3h', \sigma_s = h' \land [h/\sigma, \alpha_o/\alpha]I \land \alpha_s = \bot \land \mu_s = \text{own} \}\]

2. (semi-easy) Consider the program $\text{decr}$, which locks, decreases $x$, then unlocks. Can you verify that program with the PCM of nats? Can you think of PCMs in which it can be verified?

   - We either have to switch to integers instead of nats, or to histories PCM.
   - We can also keep a PCM of pairs $(a, b)$. The component $a$ says how much we’ve incremented, $b$ says how much we’ve decremented. But that’s basically an encoding of integers.

3. (harder) Generalize the implementation of $\text{incr}$, so that it takes an argument $k : \text{nat}$, and increments $x$ by $k$.

   Then prove that the program that iterates over the list $[k_1, k_2, \ldots, k_n]$, and forks the thread that $\text{incr}$’s over each element, upon termination, increments $x$ by $k_1 + \cdots + k_n$.

   \[
   \text{iterate } \text{nil} = \text{return}()
   \]

   \[
   \text{iterate } (x :: xs) = \text{incr}(x) || \text{iterate } xs; \text{return}()
   \]