Principles for software composition 2018/19
Exam – June 19, 2019

[Ex. 1] (1st mid-term / regular exam)
Suppose one wants to insert some measure of efficiency in the operational semantics of IMP.

1. Redefine the operational semantics of IMP commands in such a way that the transition predicate takes the form

$$\langle c, \sigma \rangle \xrightarrow{n} \sigma'$$

with the meaning that "the command c, when executed in the state σ converges to the state σ' by evaluating exactly n boolean guards."

2. Prove by rule induction that for all c, σ, σ':

$$\langle c, \sigma \rangle \to \sigma' \;\Rightarrow\; \exists n \in \mathbb{N}.\ \langle c, \sigma \rangle \xrightarrow{n} \sigma'.$$

[Ex. 2] (1st mid-term / regular exam)
Consider the $\mathrm{CPO}_\perp$ $(\wp(\mathbb{N}), \subseteq)$ and the function $f : \wp(\mathbb{N}) \to \wp(\mathbb{N})$ defined by:

$$f(X) \stackrel{\text{def}}{=} \{y \in \mathbb{N} \mid \exists a, b \in X.\ a \leq y \leq b\}$$

1. Prove that f is monotone.
2. Prove that f is continuous.

[Ex. 3] (1st mid-term)
Let us call a *repetition* any list where the same value occurs in all positions of the list. Write a Haskell function `decompose` that takes a list `xs` and returns the list of repetitions in `xs`. For example, `decompose [1,1,1,2,2,2,1,1,3]` must return the list `[[1,1,1],[2,2,2],[1,1],[3]]`.

[Ex. 4] (1st mid-term)
Consider the HOFL terms

$$t \stackrel{\text{def}}{=} \textbf{rec}\ f.\ \lambda x.\ \textbf{if}\ x\ \textbf{then}\ (x-1, f\ (x-1))\ \textbf{else}\ (x+1, f\ (x+1))$$
$$s \stackrel{\text{def}}{=} \textbf{rec}\ g.\ \lambda y.\ \textbf{if}\ y\ \textbf{then}\ g\ (y-1)\ \textbf{else}\ (y+1, \textbf{fst}(g\ (y+1)))$$

1. Find the principal type of t, if it exists.
2. Find the principal type of s, if it exists.