The problems from
Home Assignment 2
for Programming Theory (TDDA 43)
related to the denotational semantics

Version 1.2

The numbers in parentheses refer to the old version of our textbook.
The home assignment will be completed by a problem on lambda calculus and combinatory logic.

A. Exercise 5.7 (4.7) from the book. (Comparing functions from **State** $\hookrightarrow$ **State**).

B. 1. Complete (in two ways) the following definition of functions g_n: **State** $\hookrightarrow$ **State**

$$g_n s = \begin{cases} s[\mathtt{x} \mapsto \ldots] & \text{if } \; s\,\mathtt{y} < n \\ \underline{\text{undef}} & \text{if } \; s\,\mathtt{y} \geq n \end{cases}$$

(where $n \in \mathbb{Z}$, i.e. n may be any integer) so that the set $Y = \{\, g_n \mid n \in \mathbb{Z} \,\}$ is:

(a) a chain,

(b) not a chain

in the ccpo (**State** $\hookrightarrow$ **State**, $\sqsubseteq$). Explain that the set is indeed a chain, respectively not a chain.

2. Consider the set Y above that is a chain. What is the least upper bound $\bigsqcup Y$ of Y? Add one element g to Y so that $Y \cup \{g\}$ does not have an upper bound. Add one element h to Y so that $Y \cup \{h\}$ is not a chain, but has a least upper bound. Can you find an h such that the least upper bounds of Y and of $Y \cup \{h\}$ are (i) the same, (ii) not the same?

C. Let $(D, \sqsubseteq)$, $(D', \sqsubseteq)$ be partially ordered sets and $f\colon D \to D'$ be a function such that for each non-empty chain $Y \subseteq D$ the least upper bound $\bigsqcup'\{\, fd \mid d \in Y \}$ exists and

$$\bigsqcup{}'\{\, fd \mid d \in Y \} = f\left(\bigsqcup Y\right).$$

Show that f is monotone.

Hint. You have to show that $fd_1 \sqsubseteq fd_2$ for any $d_1 \sqsubseteq d_2$. Consider $Y = \{d_1, d_2\}$.

D. Which of the functions given in Exercise 5.28 (4.28) are monotone?

E. Exercise 6.9 (4.70) from the textbook. (Add call by value arguments to the language **Proc**; modify its denotational semantics to describe this extension.) Part of the task is already done, by defining the semantics of the procedure call by:

$$\begin{aligned} \mathcal{S}_{\mathrm{ds}}[\![\texttt{call}\ p(a_1, a_2)]\!]\ env_V\ env_P\ sto \ &= \ env_P\ p\ (\mathcal{A}[\![a_1]\!]s,\ \mathcal{A}[\![a_2]\!]s)\ sto \\ \text{where}\ \ s \ &= \ \text{lookup}\ env_V\ sto \end{aligned}$$

It remains to describe the semantics of procedure declarations.