Practice Problems – Operational Semantics

1. Recall the language IMP from class:

   \[
   a ::= n \mid X \mid a_0 + a_1 \mid a_0 - a_1 \mid a_0 * a_1 \\
   b ::= bv \mid a_0 = a_1 \mid a_0 \leq a_1 \mid \neg b \mid a_0 \land a_1 \mid a_0 \lor b_1 \\
   c ::= \text{skip} \mid X := a \mid \text{if } b \text{ then } c_0 \text{ else } c_1 \mid \text{while } b \text{ do } c \\
   bv ::= \text{true} \mid \text{false}
   \]

Suppose we extend the language with a C-style for loop:

   \[
   c ::= \cdots \mid \text{for}(c_0; b; c_1) \ c_2
   \]

Write down big-step operational semantics for “for.” You may not use “while” in the hypotheses of your “for” rules. \textit{Hint}: the “skip” command may come in handy.

2. Here is the lambda calculus, extended with integers, and its semantics:

\[
\begin{align*}
\text{e} & ::= v \mid x \mid e \ e \\
\text{v} & ::= n \mid \lambda x . e \\
\text{Beta} & \quad (\lambda x . e_1) v_2 \rightarrow e_1[x \mapsto v_2] \\
\text{Left} & \quad e_1 \rightarrow e_1' \\
\text{Right} & \quad e_2 \rightarrow e_2'
\end{align*}
\]

Draw derivations showing that the following reductions hold:

(a) \((\lambda x . 42) 13 \rightarrow 42\)
(b) \(((\lambda x . (\lambda y . y)) (\lambda z . z)) \rightarrow (\lambda y . y) (\lambda z . z)\)
(c) \((\lambda x . ((\lambda x . \lambda y . x) 42)) \rightarrow (\lambda x . x) (\lambda y . 42)\)

3. Write down big-step semantics for lambda calculus that are equivalent to the rules above (for terminating programs).

4. Draw a derivation of the following in your big-step semantics: \((\lambda x . x) ((\lambda x . \lambda y . x) 42) \rightarrow (\lambda y . 42)\)