Assignment on Operational Semantics

All the problems use the language *HW* whose syntax and small-step operational semantics are given in the appendix at the end of this document.

1. Suppose $(c_1\,\mathbf{;}\,c_2, \sigma) \longrightarrow^* (c_2, \sigma')$. Show that it is not necessarily the case that $(c_1, \sigma) \longrightarrow^* (\mathbf{skip}, \sigma')$.

2. In this problem we add the expressions $x{+}{+}$ and ${+}{+}x$ to the language *HW*. We extend the syntax as follows:

$$(\mathit{IExp}) \quad e \quad ::= \quad \ldots \mid x{+}{+} \mid {+}{+}x$$

The expression $x{+}{+}$ returns the value of the variable x and then increments x (i.e. updates the value of x to be one greater than the old value). Its semantics can be formalized as follows:

$$\frac{\sigma(x) = \lfloor \mathbf{n} \rfloor}{(x{+}{+}, \sigma) \longrightarrow (\mathbf{n}, \sigma\{x \rightsquigarrow \lfloor \mathbf{n} \rfloor + 1\})}$$

   (a) Give the small-step operational semantics rule for ${+}{+}x$, which increments x and then returns the result.

   (b) Give the full execution path for the program

$$\mathbf{while}\ x < \mathbf{2}\ \mathbf{do}\ x := (x{+}{+}) + ({+}{+}x)$$

   from the initial state $\{x \rightsquigarrow 1\}$.

3. In this problem we add "side-effecting expressions" to the language *HW*. We extend the syntax as follows:

$$(\mathit{IExp}) \quad e \quad ::= \quad \ldots \mid \mathsf{do}\ c\ \mathsf{return}\ e$$

The new expression first runs the command c and then returns the value of e. For example, $\mathsf{do}\ x := \mathbf{1}\ \mathsf{return}\ x$ will set x to 1 and return $\mathbf{1}$.

   (a) Give the small-step operational semantics rules for the new expression $\mathsf{do}\ c\ \mathsf{return}\ e$.

   (b) We define $\prec$ between two expressions as follows (here we write $\longrightarrow^*$ for zero-or-multiple steps of $\longrightarrow$):

$$\begin{aligned} & e_1 \prec e_2 \ \text{ iff} \\ & \quad \forall \sigma, \mathbf{n}_1, \mathbf{n}_2, \sigma_1, \sigma_2.\ ((e_1, \sigma) \longrightarrow^* (\mathbf{n}_1, \sigma_1)) \wedge ((e_2, \sigma) \longrightarrow^* (\mathbf{n}_2, \sigma_2)) \\ & \quad \Rightarrow\ (\lfloor \mathbf{n}_1 \rfloor \ < \ \lfloor \mathbf{n}_2 \rfloor) \end{aligned}$$

For each of the following properties, does it hold? If your answer is yes, just say yes. If your answer is no, give a counterexample (that is, in (i) and (ii), instantiate e_1, e_2 so that the relation $\prec$ breaks; in (iii), instantiate e so that $e \prec e$; in (iv), instantiate e_1, e_2, e_3 so that the implication fails) and explain why the property does not hold at your example.

i. $\forall e_1, e_2.\ (e_1 + e_2) \prec (e_1 + e_2 + \mathbf{1})$

ii. $\forall e_1, e_2.\ (e_1 + e_2) \prec (e_2 + e_1 + \mathbf{1})$

iii. $\forall e.\ \neg(e \prec e)$

iv. $\forall e_1, e_2, e_3.\ (e_1 \prec e_2) \wedge (e_2 \prec e_3) \ \Rightarrow\ (e_1 \prec e_3)$

Appendix: The Language *HW*

Syntax:

$$\begin{array}{rcll} (\mathit{IExp}) & e & ::= & \mathbf{n} \mid x \mid e+e \mid \ldots \\ (\mathit{BExp}) & b & ::= & \mathbf{true} \mid \mathbf{false} \mid e<e \mid \ldots \\ (\mathit{Comm}) & c & ::= & \mathbf{skip} \mid x:=e \mid c\,\mathbf{;}\,c \mid \mathbf{if}\ b\ \mathbf{then}\ c\ \mathbf{else}\ c \mid \mathbf{while}\ b\ \mathbf{do}\ c \\ (\mathit{State}) & \sigma & \in & \mathit{Var} \to \mathit{Nat} \end{array}$$

Small-step operational semantics rules:

$$\frac{(e_1,\sigma) \longrightarrow (e_1',\sigma')}{(e_1+e_2,\sigma) \longrightarrow (e_1'+e_2,\sigma')} \qquad \frac{(e_2,\sigma) \longrightarrow (e_2',\sigma')}{(\mathbf{n}+e_2,\sigma) \longrightarrow (\mathbf{n}+e_2',\sigma')}$$

$$\frac{\lfloor \mathbf{n}_1 \rfloor \ + \ \lfloor \mathbf{n}_2 \rfloor \ = \ \lfloor \mathbf{n} \rfloor}{(\mathbf{n}_1+\mathbf{n}_2,\sigma) \longrightarrow (\mathbf{n},\sigma)} \qquad \frac{\sigma(x) = \lfloor \mathbf{n} \rfloor}{(x,\sigma) \longrightarrow (\mathbf{n},\sigma)}$$

$$\frac{(e_1,\sigma) \longrightarrow (e_1',\sigma')}{(e_1<e_2,\sigma) \longrightarrow (e_1'<e_2,\sigma')} \qquad \frac{(e_2,\sigma) \longrightarrow (e_2',\sigma')}{(\mathbf{n}<e_2,\sigma) \longrightarrow (\mathbf{n}<e_2',\sigma')}$$

$$\frac{\lfloor \mathbf{n}_1 \rfloor \ < \ \lfloor \mathbf{n}_2 \rfloor}{(\mathbf{n}_1<\mathbf{n}_2,\sigma) \longrightarrow (\mathbf{true},\sigma)} \qquad \frac{\lfloor \mathbf{n}_1 \rfloor \ \geq \ \lfloor \mathbf{n}_2 \rfloor}{(\mathbf{n}_1<\mathbf{n}_2,\sigma) \longrightarrow (\mathbf{false},\sigma)}$$

$$\frac{(e,\sigma) \longrightarrow (e',\sigma')}{(x:=e,\sigma) \longrightarrow (x:=e',\sigma')} \qquad \frac{}{(x:=\mathbf{n},\sigma) \longrightarrow (\mathbf{skip},\sigma\{x \rightsquigarrow \lfloor \mathbf{n} \rfloor\})}$$

$$\frac{(c_0,\ \sigma) \longrightarrow (c_0',\ \sigma')}{(c_0\,\mathbf{;}\,c_1,\ \sigma) \longrightarrow (c_0'\,\mathbf{;}\,c_1,\ \sigma')} \qquad \frac{}{(\mathbf{skip}\,\mathbf{;}\,c_1,\ \sigma) \longrightarrow (c_1,\ \sigma)}$$

$$\frac{(b,\sigma) \longrightarrow (b',\sigma')}{(\mathbf{if}\ b\ \mathbf{then}\ c_0\ \mathbf{else}\ c_1,\sigma) \longrightarrow (\mathbf{if}\ b'\ \mathbf{then}\ c_0\ \mathbf{else}\ c_1,\sigma')}$$

$$\frac{}{(\mathbf{if\ true\ then}\ c_0\ \mathbf{else}\ c_1,\ \sigma) \longrightarrow (c_0,\ \sigma)}$$

$$\frac{}{(\mathbf{if\ false\ then}\ c_0\ \mathbf{else}\ c_1,\ \sigma) \longrightarrow (c_1,\ \sigma)}$$

$$\frac{}{(\mathbf{while}\ b\ \mathbf{do}\ c,\sigma) \longrightarrow (\mathbf{if}\ b\ \mathbf{then}\ (c\,\mathbf{;}\,\mathbf{while}\ b\ \mathbf{do}\ c)\ \mathbf{else\ skip},\sigma)}$$

Zero-or-more steps:

$$\begin{array}{lll} (c,\sigma) \to^0 (c',\sigma') & \text{iff} & (c,\sigma) = (c',\sigma') \\ (c,\sigma) \to^{k+1} (c',\sigma') & \text{iff} & \exists c'',\sigma''.\ (c,\sigma) \to (c'',\sigma'') \wedge (c'',\sigma'') \to^k (c',\sigma') \\ (c,\sigma) \to^* (c',\sigma') & \text{iff} & \exists k.\ (c,\sigma) \to^k (c',\sigma') \end{array}$$