Assignment 2

For your reference, we have given the syntax and the small-step semantics of the language *HW* in Figure 1.

1. Consider the following λ-expressions:

$$(\lambda f.\, \lambda x.\, f\,(f\,x))\,(\lambda b.\, \lambda x.\, \lambda y.\, b\,y\,x)\,(\lambda z.\, \lambda w.\, z)$$
$$(\lambda d.\, d\,d)\,(\lambda f.\, \lambda x.\, f\,(f\,x))$$
$$(\lambda x.\, x\,(\lambda t.\,(\lambda s.\, s)\,t))(\lambda y.\,(\lambda z.\, z\,z\,z)(y\,(\lambda x.\, \lambda y.\, x)))(\lambda t.\, t)$$

   For each of these expressions:

   (a) Give the normal order reduction sequence from this expression to its normal form.
   (b) Indicate the point where in the reduction sequence the first cannonical form is reached.
   (c) Give the eager evaluation sequence.

   You can find the reduction rules in the lecture notes. Be careful to rename bound variables when necessary.

2. For the following program in the language *HW*

$$\mathbf{while}\ x < \mathbf{4}\ \mathbf{do}\ x := x + \mathbf{2}$$

   write down all the evaluation steps with the initial state $\sigma = \{x \rightsquigarrow 1\}$.

3. Consider adding the expressions x++ and ++x to the language *HW*.

   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 full execution path for the program $x := (x{+}{+}) + (x{+}{+})$ from the initial state $\{x \rightsquigarrow 2\}$.
   (b) Give the operational semantics rule for ++x, which increments x and then returns the result.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Figure 1: The language *HW*.