Elements of Programming Languages
Tutorial 2: Substitution and alpha-equivalence
Solution notes

1. **Evaluation**

   (a) • $(\lambda x{:}\texttt{int}.\ x)\ 1$

   $$\frac{\overline{\lambda x{:}\texttt{int}.\ x \Downarrow \lambda x{:}\texttt{int}.\ x} \quad \overline{1 \Downarrow 1} \quad \overline{1 \Downarrow 1}}{(\lambda x{:}\texttt{int}.\ x)\ 1 \Downarrow 1}$$

   • $(\lambda x{:}\texttt{int}.\ x+1)\ 42$

   $$\frac{\overline{\lambda x{:}\texttt{int}.\ x+1 \Downarrow \lambda x{:}\texttt{int}.\ x+1} \quad \overline{42 \Downarrow 42} \quad \dfrac{\overline{42\Downarrow 42}\quad \overline{1\Downarrow 1}}{42+1\Downarrow 43}}{(\lambda x{:}\texttt{int}.\ x+1)\ 42 \Downarrow 43}$$

   • $(\lambda x{:}\texttt{int} \to \texttt{int}.\ x)\ (\lambda x{:}\texttt{int}.\ x)\ 1$ Type annotations elided.

   $$\frac{\dfrac{\overline{\lambda x.\ x \Downarrow \lambda x.\ x}\quad \overline{\lambda x.\ x \Downarrow \lambda x.\ x}\quad \overline{\lambda x.\ x \Downarrow \lambda x.\ x}}{(\lambda x.\ x)\ (\lambda x.\ x)\Downarrow \lambda x.\ x}\quad \overline{1\Downarrow 1}}{((\lambda x.\ x)\ (\lambda x.\ x))\ 1 \Downarrow 1}$$

   • $(\star)$ $((\lambda f{:}\texttt{int} \to \texttt{int}.\ \lambda x{:}\texttt{int}.f\ (f\ x))\ (\lambda x{:}\texttt{int}.\ x+1))\ 42$ Type annotations elided.

   $$\frac{\dfrac{\overline{(\lambda f.\ \lambda x.f\ (f\ x)) \Downarrow (\lambda f.\ \lambda x.f\ (f\ x))}\quad \overline{\lambda x.\ x+1 \Downarrow \lambda x.\ x+1}}{(\lambda f.\ \lambda x.f\ (f\ x))\ (\lambda x.\ x+1) \Downarrow \lambda x.(\lambda x.\ x+1)((\lambda x.\ x+1)x)}\quad \overline{42\Downarrow 42}\quad \dfrac{\vdots}{(\lambda x.\ x+1)((\lambda x.\ x+1)42)\Downarrow 44}}{((\lambda f.\ \lambda x.f\ (f\ x))\ (\lambda x.\ x+1))\ 42 \Downarrow 44}$$

   where

   $$\frac{\overline{\lambda x.\ x+1 \Downarrow \lambda x.\ x+1}\quad \dfrac{\overline{\lambda x.\ x+1\Downarrow \lambda x.x+1}\quad \overline{42\Downarrow 42}\quad \overline{42+1\Downarrow 43}}{(\lambda x.\ x+1)42\Downarrow 43}\quad \overline{43+1\Downarrow 44}}{(\lambda x.\ x+1)((\lambda x.\ x+1)42)\Downarrow 44}$$

   (b) If $e_1 : \tau$ then we can define $\texttt{let}\ x = e_1\ \texttt{in}\ e_2$ as $(\lambda x{:}\tau.\ e_2)\ e_1$. The evaluation rule for `let` can be emulated as follows:

   $$\frac{e_1\Downarrow v_1 \quad e_2[v_1/x]\Downarrow v}{\texttt{let}\ x = e_1\ \texttt{in}\ e_2 \Downarrow v} \Longrightarrow \frac{\overline{\lambda x{:}\tau.e_2 \Downarrow \lambda x{:}\tau.\ e_2}\quad e_1\Downarrow v_1\quad e_2[v_1/x]\Downarrow v}{(\lambda x{:}\tau.\ e_2)\ e_1\Downarrow v}$$

2. **Typechecking**

   (a) • `Int => Int`

   ```
   {x: Int => x}
   ```

   • `Int => Boolean => Int`

   ```
   {x: Int => {y: Boolean => x}}
   ```

   • `(Int => Boolean => String) => (Int => Boolean) => (Int => String)`

```
{x: (Int => Boolean => String) =>
  {y: (Int => Boolean) =>
    {z: Int => x(z)(y(z))}}}
```

(b)

- $(\lambda x{:}\texttt{int}.\ x)\ 1$

$$\dfrac{\dfrac{\dfrac{}{x : \texttt{int} \vdash x : \texttt{int}}}{\vdash \lambda x{:}\texttt{int}.\ x : \texttt{int} \to \texttt{int}} \quad \dfrac{}{\vdash 1 : \texttt{int}}}{\vdash (\lambda x{:}\texttt{int}.\ x)\ 1 : \texttt{int}}$$

- $(\lambda x{:}\texttt{int}.\ x + 1)\ 42$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{x{:}\texttt{int} \vdash x : \texttt{int}} \quad \dfrac{}{x{:}\texttt{int} \vdash 1 : \texttt{int}}}{x : \texttt{int} \vdash x + 1 : \texttt{int}}}{\vdash \lambda x{:}\texttt{int}.\ x + 1 : \texttt{int} \to \texttt{int}} \quad \dfrac{}{\vdash 42 : \texttt{int}}}{\vdash (\lambda x{:}\texttt{int}.\ x + 1)\ 42 : \texttt{int}}$$

- $(\lambda x{:}\texttt{int} \to \texttt{int}.\ x)\ (\lambda x{:}\texttt{int}.\ x)$

$$\dfrac{\dfrac{\dfrac{}{x{:}\texttt{int} \to \texttt{int} \vdash x : \texttt{int} \to \texttt{int}}}{\vdash (\lambda x{:}\texttt{int} \to \texttt{int}.\ x) : (\texttt{int} \to \texttt{int}) \to (\texttt{int} \to \texttt{int})} \quad \dfrac{\vdots}{\vdash \lambda x{:}\texttt{int}\ x : \texttt{int} \to \texttt{int}}}{\vdash (\lambda x{:}\texttt{int} \to \texttt{int}.\ x)\ (\lambda x{:}\texttt{int}.\ x) : \texttt{int} \to \texttt{int}}$$

- $(\lambda x{:}\tau.\ x\ x)$ **This expression cannot be typed. There is no way to choose τ so that the following derivation can be completed:**

$$\dfrac{\dfrac{\dfrac{??}{x{:}\tau \vdash x : \tau_1 \to \tau_2} \quad \dfrac{??}{x{:}\tau \vdash x{:}\tau_1}}{x{:}\tau \vdash x\ x : \tau_2}}{\vdash \lambda x{:}\tau.\ x\ x : \tau_2}$$

For if $\tau = \tau_1$ then we would also have to have $\tau = \tau_1 \to \tau_2$, i.e. $\tau_1 = \tau_1 \to \tau_2$ which is not possible if equality is structural.

3. **Alpha-equivalence for $\mathsf{L}_{\mathsf{Lam}}$**

(a) The missing rules are:

$\boxed{e_1 \equiv_\alpha e_2}$

$$\dfrac{e \equiv_\alpha e' \quad e_1 \equiv_\alpha e_1' \quad e_1 \equiv_\alpha e_1'}{\texttt{if}\ e\ \texttt{then}\ e_1\ \texttt{else}\ e_2 \equiv_\alpha \texttt{if}\ e'\ \texttt{then}\ e_1'\ \texttt{else}\ e_2'}$$

$$\dfrac{e_1(x{\leftrightarrow}z) \equiv_\alpha e_2(y{\leftrightarrow}z) \quad z \notin FV(e_1, e_2)}{\lambda x.e_1 \equiv_\alpha \lambda y.e_2} \qquad \dfrac{e_1 \equiv_\alpha e_1' \quad e_1 \equiv_\alpha e_1'}{e_1\ e_2 \equiv_\alpha e_1'\ e_2'}$$

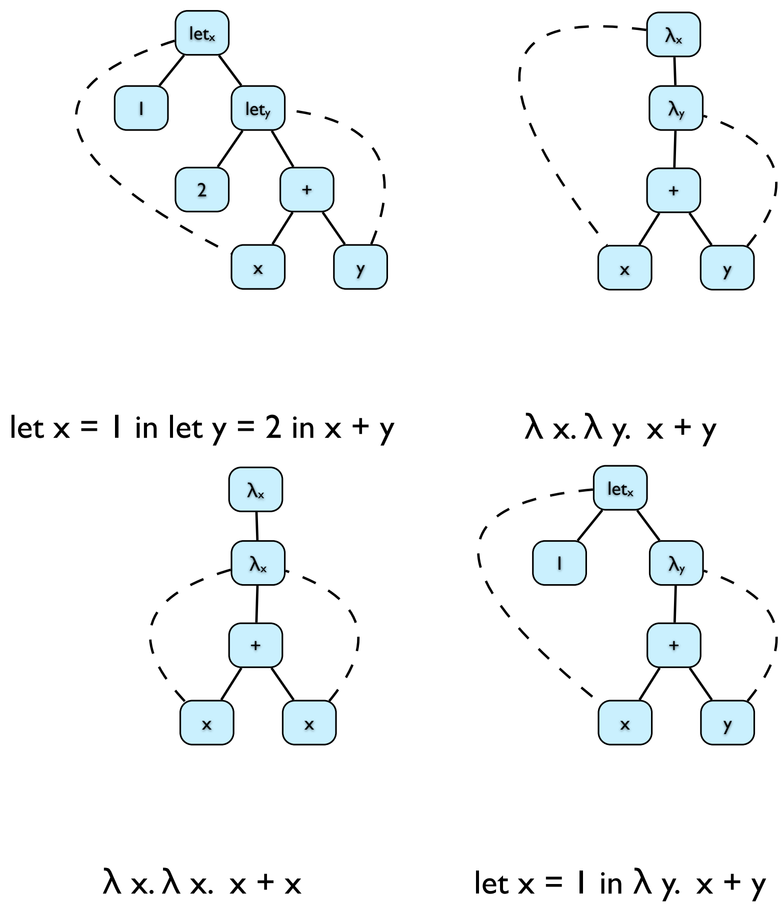
Point this out: To be precise, we should also extend FV as follows:

$$\begin{array}{rcl} FV(\lambda x{:}\tau.\ e) & = & FV(e) - \{x\} \\ FV(e_1\ e_2) & = & FV(e_1) \cup FV(e_2) \end{array}$$

(b) Which of the following alpha-equivalence relationships hold?

$\texttt{if true then}\ y\ \texttt{else}\ z$	$\equiv_\alpha$	y	FALSE
$\texttt{let}\ x = y\ \texttt{in}\ (\texttt{if}\ x\ \texttt{then}\ y\ \texttt{else}\ z)$	$\equiv_\alpha$	$\texttt{let}\ z = y\ \texttt{in}\ (\texttt{if}\ x\ \texttt{then}\ y\ \texttt{else}\ z)$	FALSE
$\lambda x.\ (\texttt{let}\ y = x\ \texttt{in}\ y + y)$	$\equiv_\alpha$	$\lambda x.\ (\texttt{let}\ x = x\ \texttt{in}\ x + x)$	TRUE

(c) The pictures should be as follows:



4. ($\star$) **Naive substitution and variable capture**

(a)

$$
\begin{aligned}
(\lambda y.\ \lambda z.\ ((x + y) + z))[y \times z/x] &= \lambda y.\ \lambda z.\ (((y \times z) + y) + z) \\
(\texttt{if } x == y \texttt{ then } \lambda z.x \texttt{ else } \lambda x.x)[z/x] &= \texttt{if } z == y \texttt{ then } \lambda z.z \texttt{ else } \lambda x.z
\end{aligned}
$$

(b)

$$
\begin{aligned}
\lambda y.\ \lambda z.\ ((x + y) + z) &\equiv_\alpha \lambda a.\ \lambda b.\ ((x + a) + b) \\
\texttt{if } x == y \texttt{ then } \lambda z.x \texttt{ else } \lambda x.x &\equiv_\alpha \texttt{if } x == y \texttt{ then } \lambda c.x \texttt{ else } \lambda d.d
\end{aligned}
$$

(c)

$$
\begin{aligned}
(\lambda a.\ \lambda b.\ ((x + a) + b))[y \times z/x] &= \lambda a.\ \lambda b.\ (((y \times z) + a) + b) \\
(\texttt{if } x == y \texttt{ then } \lambda c.x \texttt{ else } \lambda d.d)[z/x] &= \texttt{if } z == y \texttt{ then } \lambda c.z \texttt{ else } \lambda d.d
\end{aligned}
$$

Illustrate that the substitutions performed without α-conversion lead to variable capture, and different binding structure from those performed after α-converting to fresh names.