Solutions

This test consists of three exercises. *Explain your answers*. The available points for each item are written in the margin.

(8) **1.** Reduce the λ-term

$$(\lambda xy.\, y)\ (\lambda x.\, x)\ (\lambda fgx.\, f\ (g\ x))\ (\lambda x.\, x)\ (\lambda xy.\, x)\ z$$

to normal form using the leftmost innermost reduction strategy.

Solution.

$$\begin{aligned}
&(\lambda xy.\, y)\ (\lambda x.\, x)\ (\lambda fgx.\, f\ (g\ x))\ (\lambda x.\, x)\ (\lambda xy.\, x)\ z\\
&= \underline{(\lambda xy.\, y)\ (\lambda x.\, x)}\ (\lambda fgx.\, f\ (g\ x))\ (\lambda x.\, x)\ (\lambda xy.\, x)\ z\\
&\to_\beta \underline{(\lambda y.\, y)\ (\lambda fgx.\, f\ (g\ x))}\ (\lambda x.\, x)\ (\lambda xy.\, x)\ z\\
&\to_\beta \underline{(\lambda fgx.\, f\ (g\ x))\ (\lambda x.\, x)}\ (\lambda xy.\, x)\ z\\
&\to_\beta (\lambda gx.\, \underline{(\lambda x.\, x)\ (g\ x)})\ (\lambda xy.\, x)\ z\\
&\to_\beta \underline{(\lambda gx.\, g\ x)\ (\lambda xy.\, x)}\ z\\
&\to_\beta (\lambda x.\, \underline{(\lambda xy.\, x)\ x})\ z\\
&\to_\beta \underline{(\lambda xy.\, x)\ z}\\
&\to_\beta \lambda y.\, z
\end{aligned}$$

(8) **2.** Consider the Haskell functions

```
length []      =  0
length (x:xs)  =  length xs + 1

rev []      =  []
rev (x:xs)  =  rev xs ++ [x]
```

Prove by induction that

$$\texttt{length}\ (\texttt{rev}\ xs) = \texttt{length}\ xs$$

for all lists xs. You may use the equation

$$\texttt{length}\ (xs\ \texttt{++}\ ys) = \texttt{length}\ xs + \texttt{length}\ ys \qquad (\star)$$

Solution. We use induction over xs to show the property

$$P(xs) = (\texttt{length}\ (\texttt{rev}\ xs) = \texttt{length}\ xs)$$

- **Base Case** ($xs = \texttt{[]}$). $P(\texttt{[]})$ is shown by the derivation:

$$\texttt{length}\ (\texttt{rev []}) = \texttt{length}\ (\texttt{[]}) = 0$$

- **Step Case** ($xs = z : zs$). The IH is $P(zs) = (\texttt{length}\ (\texttt{rev}\ zs) = \texttt{length}\ zs)$. $P(z : zs)$ is shown by the derivation:

$$
\begin{aligned}
\texttt{length}\ (\texttt{rev}\ (z : zs)) &= \texttt{length}\ (\texttt{rev}\ zs\ \texttt{++}\ [z]) \\
&= \texttt{length}\ (\texttt{rev}\ zs) + \texttt{length}\ [z] && \text{by } (\star) \\
&\overset{\mathsf{IH}}{=} \texttt{length}\ zs + \texttt{length}\ [z] \\
&= \texttt{length}\ zs + 1 \\
&= \texttt{length}\ (z : zs)
\end{aligned}
$$

(9) **3.** Use type checking to prove that **let** $add = (\lambda xy.\, x + y)$ **in if** b **then** $add\ z\ 1$ **else** $add\ z\ 0$ has the type Int with respect to the environment $E = P \cup \{b :: \mathsf{Bool}, z :: \mathsf{Int}\}$.

Solution.

1	$x :: \mathsf{Int}$	assumption
2	$y :: \mathsf{Int}$	assumption
3	$(+) :: \mathsf{Int} \to \mathsf{Int} \to \mathsf{Int}$	ins E
4	$(+)\ x :: \mathsf{Int} \to \mathsf{Int}$	app 3, 1
5	$x + y :: \mathsf{Int}$	app 4, 2
6	$\lambda y.\, x + y :: \mathsf{Int} \to \mathsf{Int}$	abs 2–5
7	$\lambda xy.\, x + y :: \mathsf{Int} \to \mathsf{Int} \to \mathsf{Int}$	abs 1–6
8	$add :: \mathsf{Int} \to \mathsf{Int} \to \mathsf{Int}$	assumption
9	$b :: \mathsf{Bool}$	ins E
10	$z :: \mathsf{Int}$	ins E
11	$add\ z :: \mathsf{Int} \to \mathsf{Int}$	app 8, 10
12	$1 :: \mathsf{Int}$	ins E
13	$add\ z\ 1 :: \mathsf{Int}$	app 11, 12
14	$0 :: \mathsf{Int}$	ins E
15	$add\ z\ 0 :: \mathsf{Int}$	app 11, 14
16	**if** b **then** $add\ z\ 1$ **else** $add\ z\ 0 :: \mathsf{Int}$	ite 9, 13, 15
17	**let** $add = (\lambda xy.\, x + y)$ **in if** b **then** $add\ z\ 1$ **else** $add\ z\ 0 :: \mathsf{Int}$	let 7, 8–16