universität
innsbruck

Functional Programming	WS 2017/2018	LVA 703025
Test B		January 12, 2018

NAME: **MATRICULATION NUMBER:**

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

SCORE

1	2	3	total

[15] 1 Consider the Haskell function

```
foldl :: (b -> a -> b) -> b -> [a] -> b
foldl f b [] = b
foldl f b (x:xs) = foldl f (f b x) xs
```

[10] (a) Evaluate `foldl (+) 0 [1,2,3]` stepwise.

answer

```
foldl (+) 0 [1,2,3] = foldl (+) ((+) 0 1) [2,3]
                    = foldl (+) ((+) ((+) 0 1) 2) [3]
                    = foldl (+) ((+) ((+) ((+) 0 1) 2) 3) []
                    = (+) ((+) ((+) 0 1) 2) 3
                    = (+) ((+) 1 2) 3
                    = (+) 3 3
                    = 6
```

[5] (b) Implement `length` using `foldl`.

answer

One solution is `length xs = foldl (\acc _ -> acc + 1) 0 xs`, which is equivalent to `length = foldl (const . (+) 1) 0`.

[15] 2 Consider the two Haskell functions

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

and

```
replicate n x | n > 0 = x : replicate (n-1) x
replicate _ _ = []
```

Prove by mathematical induction that `length` (`replicate` n x) $= n$.

answer

Let $P(n) = ($`length` (`replicate` $n\ x) = n)$.

Base Case $(n = 0)$. We have to show $P(0)$, that is, `length` (`replicate` $0\ x) = 0$. We conclude by the following derivation

$$\begin{aligned} \texttt{length}\ (\texttt{replicate}\ 0\ x) &= \texttt{length}\ \texttt{[]} && \text{(definition of \texttt{replicate})} \\ &= 0 && \text{(definition of \texttt{length})} \end{aligned}$$

Step Case $(n = k+1)$. The induction hypothesis (IH) is $P(k)$, that is, `length` (`replicate` $k\ x) = k$. We have to show $P(k+1)$, that is, `length` (`replicate` $(k+1)\ x) = k+1$. We conclude by the following derivation

$$\begin{aligned} \texttt{length}\ (\texttt{replicate}\ (k+1)\ x) &= \texttt{length}\ (x : \texttt{replicate}\ k\ x) && \text{(definition of \texttt{replicate})} \\ &= 1 + \texttt{length}\ (\texttt{replicate}\ k\ x) && \text{(definition of \texttt{length})} \\ &= 1 + k && \text{(IH)} \\ &= k + 1 && \text{(commutativity of } +) \end{aligned}$$

[20] 3 Consider the λ-term $t = (\lambda xyz.\, x\ z\ (y\ z))\ (\lambda xy.\, z)\ (\lambda xy.\, y)\ (\lambda xy.\, x)$.

[6] (a) Compute $\mathcal{F}(t)$. No explanation needed.

answer

$\mathcal{F}(t) = \{z\}$

[6] (b) Determine for each of the following λ-terms if it is α-equivalent to t. No explanation needed.

i. $t_1 = (\lambda zyx.\, z\ x\ (y\ x))\ (\lambda xy.\, z)\ (\lambda xy.\, y)\ (\lambda xy.\, x)$.

ii. $t_2 = (\lambda xyz.\, x\ z\ (y\ z))\ (\lambda zy.\, x)\ (\lambda xy.\, y)\ (\lambda xy.\, x)$.

iii. $t_3 = (\lambda xyz.\, x\ z\ (y\ z))\ (\lambda xy.\, z)\ (\lambda zy.\, y)\ (\lambda xy.\, x)$.

answer

i. Yes.

ii. No.

iii. Yes.

[8] (c) Reduce t to β-normal form using normal order reduction.

answer

$$
\begin{aligned}
t &= \underline{(\lambda xyz.\, x\ z\ (y\ z))\ (\lambda xy.\, z)}\ (\lambda xy.\, y)\ (\lambda xy.\, x) \\
&\to_\beta \underline{(\lambda yz'.\, (\lambda xy.\, z)\ z'\ (y\ z'))\ (\lambda xy.\, y)}\ (\lambda xy.\, x) \\
&\to_\beta \underline{(\lambda z'.\, (\lambda xy.\, z)\ z'\ ((\lambda xy.\, y)\ z'))\ (\lambda xy.\, x)} \\
&\to_\beta \underline{(\lambda xy.\, z)\ (\lambda xy.\, x)}\ ((\lambda xy.\, y)\ (\lambda xy.\, x)) \\
&\to_\beta \underline{(\lambda y.\, z)\ ((\lambda xy.\, y)\ (\lambda xy.\, x))} \\
&\to_\beta z
\end{aligned}
$$