University of Innsbruck | **Institute of Computer Science**
Test 1A | **November 26, 2010**

Functional Programming | **WS 2010/2011** | **LVA 703018**

Solutions

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

(8) **1.** Given the Haskell function

```
foldr f b []     = b
foldr f b (x:xs) = f x (foldr f b xs)
```

use *equational reasoning* to evaluate the function call `foldr (:) [] [1,2,3,4]` and give at least four intermediate steps.

Solution.

```
foldr (:) [] [1,2,3,4]  = 1 : foldr (:) [] [2,3,4]
                        = 1 : 2 : foldr (:) [] [3,4]
                        = 1 : 2 : 3 : foldr (:) [] [4]
                        = 1 : 2 : 3 : 4 : foldr (:) [] []
                        = 1 : 2 : 3 : 4 : []
                        = [1,2,3,4]
```

(8) **2.** Implement a function `nub : [a] -> [a]` that eliminates duplicates from a given list, e.g.,

`nub [1,2,3,1,3,4] = [1,2,3,4]`

It is not allowed to use any predefined functions.

Solution.

```
nub []     = []
nub (x:xs) = x : nub (filter (/= x) xs)
  where filter p []   = []
        filter p (x:xs)
          | p x       = x : filter p xs
          | otherwise = filter p xs
```

3. Consider the lambda-term $t = (\lambda yz.\, y)\ ((\lambda x.\, x)\ x)$.

(5) (a) Reduce t to normal form by applying the beta-rule repeatedly.

Solution.

$$\begin{aligned}&(\lambda yz.\, y)\ \underline{((\lambda x.\, x)\ x)}\\ \rightarrow_\beta\ &\underline{(\lambda yz.\, y)\ x}\\ \rightarrow_\beta\ &\lambda z.\, x\end{aligned}$$

(4) (b) Give the three sets $\mathcal{FV}\mathsf{ar}(t)$, $\mathcal{BV}\mathsf{ar}(t)$, and $\mathcal{V}\mathsf{ar}(t)$ of the (free/bound) variables of t.

Solution.

$$\begin{aligned}\mathcal{FV}\mathsf{ar}(t) &= \{x\}\\ \mathcal{BV}\mathsf{ar}(t) &= \{x, y, z\}\\ \mathcal{V}\mathsf{ar}(t) &= \{x, y, z\}\end{aligned}$$