Functional Programming | WS 2007/2008 | LVA 703018

Solutions

1.

$$\underline{(\lambda f.(\lambda x.f\ (x\ x))\ (\lambda x.f\ (x\ x)))\ (\lambda yz.z)} \to_\beta \underline{(\lambda x.(\lambda yz.z)\ (x\ x))\ (\lambda x.(\lambda yz.z)\ (x\ x))}$$
$$\to_\beta \underline{(\lambda yz.z)\ ((\lambda x.(\lambda yz.z)\ (x\ x))\ (\lambda x.(\lambda yz.z)\ (x\ x)))}$$
$$\to_\beta \lambda z.z$$

2. *Proof.* The property to prove is

$$P(xs) = (\mathsf{sum}\ xs = \mathsf{fold}\ (+)\ 0\ xs).$$

Base Case ($\mathsf{xs} = []$). To show: $P([])$, i.e.,

$$\mathsf{sum}\ [] = \mathsf{fold}\ (+)\ 0\ [].$$

Starting from the lhs, following derivation concludes the base case:

$$\begin{aligned} \mathsf{sum}\ [] &= 0 && \text{(definition of sum)} \\ &= \mathsf{fold}\ (+)\ 0\ [] && \text{(definition of fold)} \end{aligned}$$

Step Case ($\mathsf{xs} = y :: ys$). To show: $P(ys) \to P(y :: ys)$. Therefor assume the IH $P(ys)$, i.e.,

$$\mathsf{sum}\ ys = \mathsf{fold}\ (+)\ 0\ ys.$$

It remains to show that $P(y :: ys)$. This is done as follows:

$$\begin{aligned} \mathsf{sum}\ (y :: ys) &= y + \mathsf{sum}\ ys && \text{(definition of sum)} \\ &= y + \mathsf{fold}\ (+)\ 0\ ys && \text{(IH)} \\ &= \mathsf{fold}\ (+)\ 0\ (y :: ys) && \text{(definition of fold)} \end{aligned}$$

□

3. (a) No, sum is not tail recursive since the last function call in the recursive case of its definition is to (+). Here is a tail recursive alternative:

```
let sum xs =
 let rec sum acc = function
  | [] -> acc
  | x :: xs -> sum (x + acc) xs
 in sum 0 xs
;;
```

(b) Yes, f is tail recursive since the last function call in the recursive case is to itself.