p12 q11

AD

Paper 12
Introduction to Functional Programming 2000

1. The type of foldl is $(\alpha * \beta \to \beta) \to \beta \to \alpha$ list $\to \beta$ foldr op/ has type real -> real list -> real

2. (a) `val product = foldl op* 1.0;`

   (b) `fun exists p = foldl (fn (a,b) => p(a) orelse b) false;`

   (c) `fun length l = foldl (fn (_,n) => n+1) 0 l;`

3. We use induction on the stronger statement: for all n:

$$\texttt{foldl op+ } n\ l = \texttt{foldr op+ } n\ l$$

Base case: when the list l is empty:

$$\begin{array}{lll} \texttt{foldl op+ } n\ [] & = n & \text{by definition of } \texttt{foldl} \\ & = \texttt{foldr op+ } n\ [] & \text{by definition of } \texttt{foldr} \end{array}$$

Induction Step: $l = h{::}t$

$$\begin{array}{lll} \texttt{foldl} & \texttt{op+ } n\ h{::}t & \\ & = \texttt{foldl op+ } h + n\ t & \text{by definition of } \texttt{foldl} \\ & = \texttt{foldr op+ } h + n\ t & \text{by induction hypothesis} \\ & = \texttt{foldr op+ } h + n\ t & \text{by induction hypothesis} \\ & = h + \texttt{foldr op+ } n\ t & \text{by properties of addition} \\ & = \texttt{foldr op+ } n\ h{::}t & \text{by definition of } \texttt{foldr} \end{array}$$