Cálculo de Programas

Licenciatura em Engenharia Informática

Ficha 11

1. Considere o seguinte tipo, que pode ser usado para retornar mensagens de erro em funções parciais.

   $$\textbf{data}\ \mathsf{Error}\ a = \mathsf{Error\ String} \mid \mathsf{Result}\ a$$

   (a) Implemente a respectiva instância da classe Monad.

   (b) Usando este *monad* implemente uma função $\mathsf{eval} :: [(\mathsf{String}, \mathsf{Int})] \to \mathsf{Exp} \to \mathsf{Error\ Int}$, que devolva mensagens de erro apropriadas quando aplicada ao seguinte tipo:

   $$\textbf{data}\ \mathsf{Exp} = \mathsf{Const\ Int} \mid \mathsf{Var\ String} \mid \mathsf{Plus\ Exp\ Exp} \mid \mathsf{Div\ Exp\ Exp}$$

2. Defina a função $\mathsf{sequence} :: \mathsf{Monad}\ m \Rightarrow [m\ a] \to m\ [a]$, que dada uma lista de computações monádicas, executa todas essas computações e devolve uma computação com a lista dos resultados.

3. Demonstre as seguintes propriedades da composição monádica:

   (a) $f \bullet \mathsf{return} = f$

   (b) $(f \bullet g) \bullet h = f \bullet (g \bullet h)$

4. Considere o *monad* das listas.

   (a) Defina as funções return e join no estilo *point-free*.

   (b) Demonstre que estas definições satisafazem as leis dos *monads*:

   $$\mathsf{join} \circ \mathsf{join} = \mathsf{join} \circ \mathsf{map\ join}$$
   $$\mathsf{join} \circ \mathsf{return} = \mathsf{join} \circ \mathsf{map\ return} = \mathsf{id}$$

   Assuma que $\mathsf{cat} \circ (\mathsf{wrap} \times \mathsf{id}) = \mathsf{cons}$.

5. O seguinte tipo também pode ser considerado um *monad*.

   $$\textbf{data}\ \mathsf{Tree}\ a = \mathsf{Leaf}\ a \mid \mathsf{Node}\ (\mathsf{Tree}\ a)\ (\mathsf{Tree}\ a)$$

   (a) Comece por demonstrar que é um funtor, ou seja, defina no estilo *point-free* a função $\mathsf{map_T} :: (a \to b) \to (\mathsf{Tree}\ a \to \mathsf{Tree}\ b)$ e demonstre que $\mathsf{map_T}\ f \circ \mathsf{map_T}\ g = \mathsf{map_T}\ (f \circ g)$ e $\mathsf{map_T}\ \mathsf{id} = \mathsf{id}$.

   (b) Defina as funções return e join no estilo *point-free*.