Métodos de Programação I
Exame – Recurso

17 de Fevereiro 2005

1. Calcule caso exista, justificando, o tipo da função $\langle i_1, \pi_1 \rangle$.

2. Considerando os isomorfismos bem conhecidos

   - $a : A^2 \rightarrow A \times A$
   - $b : 2 \times A \rightarrow A + A$
   - $c : A \times (B \times C) \rightarrow (A \times B) \times C$
   - $d : A \times (B + C) \rightarrow (A \times B) + (A \times C)$
   - $f : (B + C) \times A \rightarrow (B \times A) + (C \times A)$

   sintetize, justificando, o seguinte isomorfismo

   $$v : A \times (1 + X)^2 \rightarrow A + 2 \times A \times X + A \times X^2$$

3. Sejam $\mathsf{F}\ f = id + f$ e $\mathsf{G}\ f = id + id \times f$ os functores-base dos naturais e listas de naturais, respectivamente, $in_{\mathsf{F}} = [\underline{Zero}, Succ]$ e $in_{\mathsf{G}} = [\underline{Nil}, Cons]$.

   Escreva $\mathsf{geraLista} = in_{\mathsf{G}} \cdot (\mathsf{G}\ \mathsf{geraLista}) \cdot (id + \langle Succ, id \rangle) \cdot out_{\mathsf{F}}$ em Haskell com variáveis, justificando todos os passos.

4. Considere a função que agrupa elementos consecutivos iguais; quando aplicada por exemplo à lista `"aaabccdddd"` esta função deve retornar `[('a',3),('b',1),('c',2),('d',4)]`.

   (a) Defina, se possível, esta função como catamorfismo de listas e como anamorfismo de listas. Justifique caso não seja possível.

   (b) Defina em Haskell um tipo de dados para *listas não vazias* e desenhe os respectivos diagramas de catamorfismos e anamorfismos. Repita agora a alínea anterior (i.e. defina se possível a mesma função como catamorfismo e anamorfismo de listas não vazias).

   (c) Escreva a seguinte função como hilomorfismo de listas não vazias:

```
strange :: (Int, Int) -> Int
strange (n, m) | n == m = n
               | n < m = strange (n+1,m-1)
               | otherwise = strange (n,m+1)
```

5. Considere a seguinte definição de árvores binárias de procura.

```
data BTree a = V | N x (BTree a) (BTree a)
```

Considere ainda a seguinte função de cálculo do menor elemento de uma árvore.

```
minArv :: (Ord a) => BTree a -> a
minArv (N x V V) = x
minArv (N x e V) = min x (minArv e)
minArv (N x V d) = min x (minArv d)
minArv (N x e d) = min x (min (minArv e) (minArv d))
```

(a) Escreva esta definição como um hilomorfismo. Diga qual a estrutura de dados intermédia usada no cálculo do menor elemento da árvore

```
(N 4 (N 10 (N 2 V V) (N 1 V V)) (N 3 (N 5 V V) V))
```

(b) Uma árvore binária diz-se uma *heap* se qualquer caminho da raíz até uma folha é decrescente (a raíz é por isso o maior elemento da árvore). Altere o gene do catamorfismo da alína anterior de forma a calcular o menor elemento de uma *heap*.