Métodos de Programação I

2.º Ano da LMCC (7003N5) + LESI (5303O7)
Ano Lectivo de 2001/2002

Exame (época de recurso) — 16 de Setembro 2002
14h30
Salas 2201 a 2204

NB: *Esta prova consta de **8** alíneas que valem, cada uma, 2.5 valores. Responda no enunciado, preenchendo a sua identificação em todas as folhas.*

PROVA SEM CONSULTA (2 horas)

Questão 1 Demonstre a igualdade

$$(id \times \pi_2) \cdot assocr = \pi_1 \times id \tag{1}$$

onde $A \times (B \times C) \xleftarrow{assocr} (A \times B) \times C$ é uma função que conhece:

$$assocr \stackrel{\text{def}}{=} \langle \pi_1 \cdot \pi_1, \langle \pi_2 \cdot \pi_1, \pi_2 \rangle \rangle \tag{2}$$

Questão 2 Considere a seguinte função em HASKELL que calcula o quadrado de um número:

```
sq 0 = 0
sq (n+1) = 2*n+1 + sq n
```

1. Mostre que sq satisfaz a equação

$$sq \cdot in = [\underline{0}, add \cdot \langle odd, sq \rangle] \tag{3}$$

   onde $in = [\underline{0}, succ]$, $\overline{add} = (+)$ e $odd = suc \cdot add \cdot \langle id, id \rangle$.
   Sugestão: Recupere o HASKELL acima a partir da conversão da equação dada para notação com variáveis.

2. Complete as justificações do seguinte processo de cálculo que converte sq num hilomorfismo de listas, partindo de (3):

$$\begin{array}{ll}
& sq \cdot in = [\underline{0}, add \cdot \langle odd, sq \rangle] \\
\equiv & \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
& sq \cdot in = [\underline{0}, add] \cdot (id + \langle odd, sq \rangle) \\
\equiv & \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
& sq \cdot in = [\underline{0}, add] \cdot (id + odd \times sq) \cdot (id + \langle id, id \rangle) \\
\equiv & \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
& sq \cdot in = [\underline{0}, add] \cdot (id + id \times sq) \cdot (id + odd \times id) \cdot (id + \langle id, id \rangle) \\
\equiv & \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
& sq = [\underline{0}, add] \cdot (id + id \times sq) \cdot (id + \langle odd, id \rangle) \cdot out \\
\equiv & \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
& sq = [\![\, [\underline{0}, add], (id + \langle odd, id \rangle) \cdot out \,]\!] \\
\equiv & \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
& sq = (\![\, [\underline{0}, add] \,]\!) \cdot [\!((id + \langle odd, id \rangle) \cdot out)\!]
\end{array}$$

3. Escrever $[\!((id + \langle odd, id \rangle) \cdot out)\!]$ em Haskell com variáveis e descrever o resultado da aplicação deste anamorfismo ao argumento $n = 3$.

Questão 3 Dê definições para as funções f_1 a f_9 da seguinte tabela, onde se apresentam três tipos paramétricos de dados em HASKELL que são instâncias da classe `Monad`:

F a	return	(>>=)	μ
`Maybe a`	f_1	f_2	f_3
`[a]`	f_4	f_5	f_6
`Error a`	f_7	f_8	f_9

onde `data Error a = Err String | Ok a`.

Questão 4 Recorde a formulação como um hilomorfismo do algoritmo "quick sort"

```
qSort = hyloBTree inord qsep
--    = (cataBTree inord) . (anaBTree qsep)
```

que conhece da biblioteca `BTree.hs`, onde ocorrem os "genes"

```
inord = either (const []) join

qsep []    = Left ()
qsep (h:t) = Right (h,(s,l)) where (s,l) = part (<h) t
```

e as funções auxiliares

```
join(x,(l,r))=l++[x]++r

part p []                = ([],[])
part p (h:t) | p h       = let (s,l) = part p t in (h:s,l)
             | otherwise = let (s,l) = part p t in (s,h:l)
```

Pretende-se uma nova versão `qSort'` deste algoritmo que, para além de ordenar a lista argumento, lhe remove os elementos repetidos.

1. Defina `qSort'` a partir do hilomorfismo `hyloBTree inord qsep`, alterando apenas o gene `qsep`.
2. Repita a alínea anterior mudando agora apenas o gene `inord`.
3. Comente a eficiência das duas versões alternativas das alíneas anteriores, sem se esquecer de abordar a situação seguinte: `qSort' l`, para `l` tal que `nub l = [1]` e `length l = 100`.
