**Cálculo de Programas**

2.º Ano de LCC (Universidade do Minho)
Ano Lectivo de 2022/23

Exame de Recurso — 23 de Junho de 2023, 14h00–16h00
Salas E1-1.10 + E1-1.17

---

PROVA PRESENCIAL INDIVIDUAL SEM CONSULTA (2h)

**Importante** — *Ler antes de iniciar a prova:*

- *Esta prova consta de **8** questões que valem, cada uma, 2.5 valores. O tempo médio estimado para resolução de cada questão é de 15 min.*
- *Recomenda-se que os alunos leiam a prova antes de decidirem por que ordem querem responder às questões que são colocadas.*

**Questão 1** Mostre que a expressão

$$[i_{11}\,, i_{22}] \cdot (\langle f, h\rangle + \langle g, k\rangle) \tag{E1}$$

onde $i_{11} = i_1 \times i_1$ e $i_{22} = i_2 \times i_2$, simplifica em

$$\langle f + g, h + k\rangle \tag{E2}$$

---

**Questão 2** Recorde os isomorfismos

$$\begin{tikzcd} A \times (B \times C) \arrow[rr, bend left, "\mathsf{assocl}"] \arrow[rr, phantom, "\cong"] & & (A \times B) \times C \arrow[ll, bend left, "\mathsf{assocr}"] \end{tikzcd}$$

Com base na propriedade grátis (i.é natural) de um deles (derive-a), demonstre:

$$\mathsf{assocr} \cdot (id \times h) \cdot \mathsf{assocl} = id \times (id \times h) \tag{E3}$$

---

**Questão 3** Recorde a definição de guarda de um condicional de McCarthy:

$$\begin{tikzcd} A \arrow[r, "\langle p, id\rangle"] \arrow[rr, bend right, "p?"'] & 2 \times A \arrow[r, "\alpha"] & A + A \end{tikzcd} \tag{E4}$$

Sabendo que

$$\mathsf{join} \cdot \alpha = \pi_2 \tag{E5}$$

se verifica, onde $\mathsf{join} = [id\,, id]$, demonstre a igualdade

$$p \rightarrow id, id = id \tag{E6}$$

e daí

$$p \rightarrow f, f = f \tag{E7}$$

**Questão 4** Demonstre a propriedade

$$\overline{f \cdot g} \quad = \quad \overline{f \cdot ap} \cdot \overline{g} \qquad \text{(E8)}$$

sem recorrer às leis **Def-exp** e **Absorção-exp** do formulário.

---

**Questão 5** Recorde o tipo de dados $\mathsf{BTree}$:

- Árvores com informação de tipo $A$ nos nós

$$\mathsf{T} = \mathsf{BTree}\ A \qquad \left\{ \begin{array}{l} \mathsf{F}\ X = 1 + A \times X^2 \\ \mathsf{F}\ f = id + id \times f^2 \end{array} \right. \qquad \mathsf{in} = [\underline{Empty}, Node]$$

Haskell: $\mathbf{data}\ \mathsf{BTree}\ a = Empty \mid Node\ (a, (\mathsf{BTree}\ a, \mathsf{BTree}\ a))$

Pretende-se uma função $addNodes$ que some todos os nós de uma árvore de tipo $\mathsf{BTree}\ \mathbb{N}_0$. Defina-a como um catamorfismo, isto é, identifique $g$ em $addNodes = (\!|\, g \,|\!)$ e desenhe esse catamorfismo sob a forma de um diagrama.

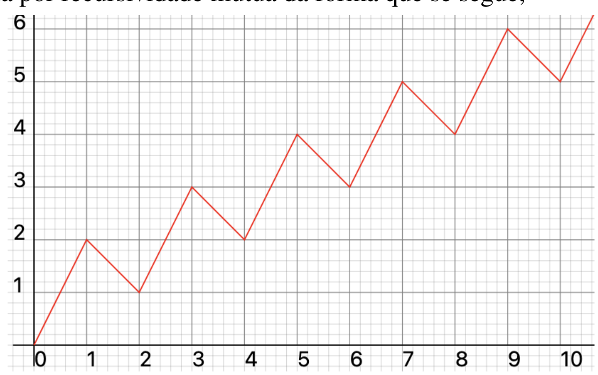
---

**Questão 6** A figura ao lado representa uma função $h$ definida por recursividade mútua da forma que se segue,

$$\left\{ \begin{array}{l} h = \mathsf{in} \cdot \mathsf{F}\ g \cdot \mathsf{out} \\ g = [\underline{1}, id] \cdot \mathsf{F}\ h \cdot \mathsf{out} \end{array} \right.$$

onde

$$\begin{array}{l} \mathsf{F}\ f = id + f \\ \mathsf{in} = [\underline{0}, \mathsf{succ}], \text{ para } \mathsf{succ}\ x = x + 1 \\ \mathsf{out} = \mathsf{in}^{\circ} \end{array}$$




Mostre, aplicando uma lei que conhece das aulas desta disciplina, que a mesma função pode ser definida à custa de um ciclo-$\mathsf{for}$, como se segue:

$$\begin{array}{l} h = \pi_1 \cdot \mathsf{for}\ loop\ (0,1)\ \mathbf{where} \\ \quad loop\ (a,b) = (1 + b, a) \end{array}$$

---

**Questão 7** Recorde o tipo de dados $\mathsf{LTree}$:

- Árvores com informação de tipo $A$ nas folhas:

$$\mathsf{T} = \mathsf{LTree}\ A \qquad \left\{ \begin{array}{l} \mathsf{F}\ X = A + X^2 \\ \mathsf{F}\ f = id + f^2 \end{array} \right. \qquad \mathsf{in} = [Leaf, Fork]$$

Haskell: $\mathbf{data}\ \mathsf{LTree}\ a = Leaf\ a \mid Fork\ (\mathsf{LTree}\ a, \mathsf{LTree}\ a)$

O catamorfismo

$$depth = (\!|\, [\underline{1}, \mathsf{succ} \cdot umax] \,|\!) \qquad \text{(E9)}$$

dá a profundidade de árvores do tipo $\mathsf{LTree}$, onde $\mathsf{succ}\ x = x + 1$ e $umax\ (a,b) = max\ a\ b$. Mostre, por absorção-cata, que a profundidade de uma árvore não é alterada quando aplica uma função $f$ a todas as suas folhas:

$$depth \cdot \mathsf{LTree}\ f = depth \qquad \text{(E10)}$$

**Questão 8** Considere o seguinte anamorfismos de listas que dá todos os sufixos de uma lista,

$$\mathsf{tails} = [\!(\, (! + \langle id, \mathsf{tail} \rangle) \cdot \mathsf{null}? \,)\!] \tag{E11}$$

onde null é o predicado que testa se uma lista é vazia ou não.

Desenhe o diagrama do anamorfismo tails e derive de (E11) uma versão *pointwise* que não recorra a nenhum dos combinadores *pointfree* estudados na disciplina.