Cálculo de Programas

2.º Ano da LCC (a) + LEI (Universidade do Minho)
Ano Lectivo de 2009/10

Avaliação Individual (Método A) —- Questão nr. 1

IDENTIFICAÇÃO DO ALUNO:

Nome: . . . . . **Número** : | | | | | |

SEM CONSULTA (15 minutos)

Questão 1 Considere as seguintes funções mutuamente inversas:

$$\begin{aligned} f &= \langle \pi_1 \cdot \pi_1, \pi_2 \times id \rangle \\ g &= \langle id \times \pi_1, \pi_2 \cdot \pi_2 \rangle \end{aligned}$$

1. Identifique, justificadamente, os seus tipos.
2. Mostre que $f \cdot g = id$.

RESPOSTA:
*(continue no verso se necessário, mas **não** continue no verso de outra página)*

Cot.:

Nr. do aluno: | | | | | |

Reservado aos docentes: | | |

Cálculo de Programas

2.º Ano da LCC () + LEI (Universidade do Minho)
Ano Lectivo de 2009/10

Avaliação Individual (Método A) —- Questão nr. 1

IDENTIFICAÇÃO DO ALUNO:

Nome: . . . . . **Número** : | | | | | |

SEM CONSULTA (15 minutos)

Questão 1 Seja $\gamma = \langle \pi_1 \cdot \pi_1, \pi_2 \times id \rangle$.

1. Mostre que $(f \times (g \times h)) \cdot \gamma = \gamma \cdot ((f \times g) \times h)$
2. Qual o tipo da função γ? Justifique.

RESPOSTA:
*(continue no verso se necessário, mas **não** continue no verso de outra página)*

Cot.:

Nr. do aluno: | | | | | |

Reservado aos docentes: | | |

Cálculo de Programas
2º Ano da LCC + LEI (Universidade do Minho)
Ano Lectivo de 2009/10
Avaliação Individual (Método A) —- Questão nr. 1

Identificação do Aluno:
Nome:
Número:

Sem Consulta(15 minutos)

Questão 1 - Parta da propriedade universal do produto para provar que a igualdade $(g.h) \times (i.j) = (g \times i).(h \times j)$ se verifica. Demonstre-a também usando diagramas.

RESPOSTA:

Cálculo de Programas
2º Ano da LCC + LEI (Universidade do Minho)
Ano Lectivo de 2009/10
Avaliação Individual (Método A) —- Questão nr. 1

Identificação do Aluno:
Nome:
Número:

Sem Consulta(15 minutos)

Questão 1 - Considere a seguinte declaração de um tipo de dados em Haskell:

```
data From a = First a | Succ (From a)
```

que induz a definição do isomorfismo

```
inFrom=[First,Succ]
```

Desenhe o diagrama que explica a a construção de `inFrom`. Calcule a sua inversa (versão "pointwise") resolvendo a equação

```
outFrom . inFrom = id
```

em ordem a `outFrom`.

Resposta:

Cálculo de Programas

2.º Ano da LCC e da LEI (Universidade do Minho)
Ano Lectivo de 2009/10 (Turno TP-3)

Avaliação Individual (Método A) —- Questão nr. 1

IDENTIFICAÇÃO DO ALUNO:

Nome: . . . . . **Número** : | | | | | |

SEM CONSULTA (15 minutos)

Questão 1 É dada a equação $[id\ , i_2] = mu$.

1. Calcule, resolvendo a equação dada em ordem mu, a seguinte implementação em Haskell:

$$mu :: Either\ (Either\ a\ b)\ b \rightarrow Either\ a\ b$$
$$mu\ (Left\ x) = x$$
$$mu\ (Right\ y) = Right\ y$$

   desenhando um diagrama explicativo do tipo de mu.

2. Mostre (por cálculo analítico) que mu satisfaz a propriedade

$$mu \cdot mu = mu \cdot (mu + id)$$

RESPOSTA:
*(continue no verso se necessário, mas **não** continue no verso de outra página)*

Cot.:

Nr. do aluno: | | | | | |

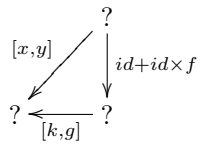
Reservado aos docentes: | | |

Cálculo de Programas
2º Ano da LCC + LEI (Universidade do Minho)
Ano Lectivo de 2009/10
Avaliação Individual (Método A) —- Questão nr. 1

Identificação do Aluno:
Nome:
Número:

Sem Consulta(15 minutos)

Questão 1 - Complete os "?" no diagrama



e resolva a equação implicita em ordem a x e a y.

Resposta:

Cálculo de Programas

2.º Ano da LCC e da LEI (Universidade do Minho)
Ano Lectivo de 2009/10

Turno TP-1: Avaliação Individual (Método A) —- Questão nr. 2

IDENTIFICAÇÃO DO ALUNO:

Nome: . . . . . **Número** : | | | | | |

SEM CONSULTA (15 minutos)

Questão 1 Demonstre a seguinte propriedade do combinador condicional de McCarthy

$$p \to (p \to a, b), (p \to c, d) \quad = \quad p \to a, d \tag{1}$$

sabendo que

$$(p? + p?) \cdot p? \quad = \quad (i_1 + i_2) \cdot p? \tag{2}$$

se verifica.

RESPOSTA:
*(continue no verso se necessário, mas **não** continue no verso de outra página)*

Cot.:

Nr. do aluno: | | | | | |

Reservado aos docentes: | | |

Cálculo de Programas

2.º Ano da LCC e da LEI (Universidade do Minho)
Ano Lectivo de 2009/10

Avaliação Individual (Método A, Turno TP-2) —- Questão nr. 2

IDENTIFICAÇÃO DO ALUNO:

Nome: . . . . . **Número** : | | | | | |

SEM CONSULTA (15 minutos)

Questão 1 Demonstre a seguinte propriedade do combinador condicional de McCarthy

$$(\neg \cdot p) \to f, g \quad = \quad p \to g, f \tag{1}$$

sabendo que é válida a propriedade

$$(\neg \cdot p)? \quad = \quad coswap \cdot (p?) \tag{2}$$

onde $coswap = [i_2 \, , i_1]$.

RESPOSTA:
*(continue no verso se necessário, mas **não** continue no verso de outra página)*

Cot.:

Nr. do aluno: | | | | | |

Reservado aos docentes: | | |

Cálculo de Programas

2.º Ano da LCC e da LEI (Universidade do Minho)
Ano Lectivo de 2009/10

Turno TP-3: Avaliação Individual (Método A) —- Questão nr. 2

IDENTIFICAÇÃO DO ALUNO:

Nome: . . . . . **Número** : | | | | | |

SEM CONSULTA (15 minutos)

Questão 1 Demonstre a seguinte propriedade do combinador condicional de McCarthy

$$\langle f, (p \to g, h) \rangle \quad = \quad p \to \langle f, g \rangle, \langle f, h \rangle \tag{1}$$

sabendo que a igualdade

$$p \to k, k \quad = \quad k \tag{2}$$

se verifica.

RESPOSTA:
*(continue no verso se necessário, mas **não** continue no verso de outra página)*

Cot.:

Nr. do aluno: | | | | | |

Reservado aos docentes: | | |

Cálculo de Programas

2.º Ano da LCC e da LEI (Universidade do Minho)
Ano Lectivo de 2009/10

Avaliação Individual (Método A, Turno TP-4) —- Questão nr. 2

IDENTIFICAÇÃO DO ALUNO:

Nome: . . . . . **Número** : | | | | | |

SEM CONSULTA (15 minutos)

Questão 1 Demonstre a seguinte propriedade do combinador condicional de McCarthy

$$f \times (p \to g, h) \quad = \quad p \cdot \pi_2 \to f \times g, f \times h \tag{1}$$

sabendo que

$$\langle f, (p \to g, h) \rangle \quad = \quad p \to \langle f, g \rangle, \langle f, h \rangle \tag{2}$$

se verifica.

RESPOSTA:
*(continue no verso se necessário, mas **não** continue no verso de outra página)*

Cot.:

Nr. do aluno: | | | | | |

Reservado aos docentes: | | |

Cálculo de Programas

2.º Ano da LCC e da LEI (Universidade do Minho)
Ano Lectivo de 2009/10

Avaliação Individual (Método A, Turno TP-2) —- Questão nr. 2

IDENTIFICAÇÃO DO ALUNO:

Nome: . . . . . **Número** : | | | | | |

SEM CONSULTA (15 minutos)

Questão 1 Demonstre a seguinte propriedade do combinador condicional de McCarthy

$$(\neg \cdot p) \to g, f \quad = \quad p \to f, g \tag{1}$$

sabendo que é válida a propriedade

$$(\neg \cdot p)? \quad = \quad [i_2 \, , i_1] \cdot (p?) \tag{2}$$

RESPOSTA:
*(continue no verso se necessário, mas **não** continue no verso de outra página)*

Cot.:

Nr. do aluno: | | | | | |

Reservado aos docentes: | | |

Cálculo de Programas
2º Ano da LCC + LEI (Universidade do Minho)
Ano Lectivo de 2009/10
Avaliação Individual (Método A) TP1 —- Questão nr. 3

Identificação do Aluno:
Nome:
Número:

Sem consulta(15 minutos)

Questão 3 - Considere a seguinte função definida em Haskell:

```
h (Left (x,y))=(x,(y,(Left ())))
h (Right (x,(y,z)))= (x,(y,(Right z)))
```

- Derive uma definição *pointfree* da função h.
- Identifique o isomorfismo testemunhado por esta função, com recurso ao diagrama respectivo.

Resposta:

Cálculo de Programas
2º Ano da LCC + LEI (Universidade do Minho)
Ano Lectivo de 2009/10
Avaliação Individual (Método A) TP2 —- Questão nr. 3

Identificação do Aluno:
Nome:
Número:

Sem Consulta(15 minutos)

Questão 3 - Considere o seguinte isomorfismo:

$$A \times (1 + B) \cong (A \times B) + A$$

- Apresente uma definição *pointfree* de uma função que testemunhe este isomorfismo da direita para a esquerda, desenhando o diagrama respectivo.
- Derive uma definição *pointwise* da função anterior.

Resposta:

Cálculo de Programas

2.º Ano da LCC e da LEI (Universidade do Minho)
Ano Lectivo de 2009/10

Turno TP-3: Avaliação Individual (Método A) —- Questão nr. 3

IDENTIFICAÇÃO DO ALUNO:

Nome: . . . . . **Número** : | | | | | |

SEM CONSULTA (15 minutos)

Questão 1 Considere a função, em Haskell

$$g\ (Leaf\ a) = Leaf\ (succ\ a)$$
$$g\ (Fork\ (x, y)) = Fork\ (g\ x, g\ y)$$

definida sobre uma instância do tipo de dados

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

do qual se infere a existência da função

$$inLTree = [Leaf\ , Fork]$$

É possível mostrar que g satisfaz a equação

$$g \cdot inLTree = inLTree \cdot k \cdot (id + g \times g)$$

para uma dada função k. Calcule k.
Sugestão: retire as variáveis a, x e y à definição dada, converta-a numa igualdade *pointfree* e compare o resultado como a equação acima.

RESPOSTA:
*(continue no verso se necessário, mas **não** continue no verso de outra página)* **Cot.:**

Cálculo de Programas
2º Ano da LCC + LEI (Universidade do Minho)
Ano Lectivo de 2009/10
Avaliação Individual (Método A) TP4 —- Questão nr. 3

Identificação do Aluno:
Nome:
Número:

Sem Consulta(15 minutos)

Questão 3 - Considere a seguinte função:

$$iso = [< id, i2.! >, id \times i1]$$

- Derive uma definição em Haskell *pointwise* da função anterior.
- Apresente a lei natural da função *iso* com recurso ao diagrama respectivo.

Resposta:

Cálculo de Programas

2.º Ano da LCC Universidade do Minho)
Ano Lectivo de 2009/10

Avaliação Individual (Método A) —- Questão nr. 3

IDENTIFICAÇÃO DO ALUNO:

Nome: . . . . . **Número** : | | | | | |

SEM CONSULTA (15 minutos)

Questão 1
Mostre que $(h + g) \cdot (h' + g') = (h \cdot h') + (g \cdot g')$ e que $id \times id = id$.

RESPOSTA:
*(continue no verso se necessário, mas **não** continue no verso de outra página)* **Cot.:**

Nr. do aluno: | | | | | |

Reservado aos docentes: | | |

Cálculo de Programas
2º Ano da LCC + LEI (Universidade do Minho)
Ano Lectivo de 2009/10
Avaliação Individual (Método A) TP1 —- Questão nr. 4

Identificação do Aluno:
Nome:
Número:

Sem Consulta(15 minutos)

Questão 4 - Relembre o tipo de dados

$$LTree\ \ a = Leaf\ \ a \mid Fork\ (LTree\ \ a, LTree\ \ a)$$

e as funções seguintes definidas como catamorfismos:

$listify :: LTree\ \ a \to [a]$
$listify = (\!|[sing, cat]|\!)$
$max_{LT} :: LTree\ \ a \to N_0$

onde max_{LT} calcula o máximo das folhas da árvore e

$sing :: a \to [a]$

$sing\ \ x = [x]$

$cat :: ([a], [a]) \to [a]$

$cat(l1, l2) = l1\ \ ++\ l2$

Verifica-se que

$$max_L\ .\ listify = max_{LT} \qquad (1)$$

sendo max_L a função que calcula o maximo de uma lista.

Aplique a lei de fusão-cata para verificar a propriedade (1), podendo usar propriedades sobre listas como, por exemplo:

$$max_L\ .\ cat = max\ .\ (max_L \times max_L)$$

Cálculo de Programas

2.º Ano da LCC e da LEI (Universidade do Minho)
Ano Lectivo de 2009/10

Turno TP-2: Avaliação Individual (Método A) —- Questão nr. 4

IDENTIFICAÇÃO DO ALUNO:

Nome: ..... **Número** : | | | | | |

SEM CONSULTA (15 minutos)

Questão 1 Como sabe, $foldr\ f\ y = (\!|[\underline{y}\ , \widehat{f}]|\!)$. Pretende-se provar, usando a lei de fusão-cata, a propriedade

$$x + (foldr\ (+)\ y\ s) = foldr\ (+)\ (x + y)\ s$$

Complete a seguinte prova desse facto, em que se abrevia $add = \widehat{(+)}$:

$$\begin{array}{ll} & x + (foldr\ (+)\ y\ s) = foldr\ (+)\ (x + y)\ s \\ \equiv & \quad \{\ \ldots\ \} \\ & (x+) \cdot (foldr\ (+)\ y) = foldr\ (+)\ (x + y) \\ \equiv & \quad \{\ \ldots\ \} \\ & (x+) \cdot (\!|[\underline{y}\ , add]|\!) = (\!|[\underline{x + y}\ , add]|\!) \\ \cdots & \quad \{\ \text{... varios passos ...}\ \} \\ & \vdots \end{array}$$

Importante: Não esqueça a propriedade

$$f \cdot \underline{k} \quad = \quad \underline{f\ k} \tag{1}$$

e assuma a que se segue, válida na aritmética,

$$x + (y + z) = y + (x + z)$$

que, em notação *pointfree*, se pode escrever

$$(x+) \cdot add \quad = \quad add \cdot (id \times (x+)) \tag{2}$$

RESPOSTA:
*(continue no verso se necessário, mas **não** continue no verso de outra página)*

Cot.:

Nr. do aluno: | | | | | |

Reservado aos docentes: | | |

Cálculo de Programas

2.º Ano da LCC e da LEI (Universidade do Minho)
Ano Lectivo de 2009/10

Turno TP-03: Avaliação Individual (Método A) —- Questão nr. 4

IDENTIFICAÇÃO DO ALUNO:

Nome: .... **Número** : [| | | |]

SEM CONSULTA (15 minutos)

Questão 1 Uma das travessias possíveis de uma árvore binária de tipo

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

é a travessia em *pré-ordem* dada pelo catamorfismo

$$
\begin{array}{l}
preord :: BTree\ a \to [a] \\
preord = (\![[nil\ , (cons \cdot (id \times cat))]]\!) \\
\quad \mathbf{where}\ nil\ \_ = [\,] \\
\qquad cons\ (h, t) = h : t \\
\qquad cat\ (x, y) = x +\!\!+\ y
\end{array}
$$

Verifica-se que o comprimento de uma travessia é sempre o número de nós da árvore, isto é, neste caso:

$$\mathsf{length} \cdot preord = count \tag{1}$$

onde length é a função que conhece (listas) e $count$ é o catamorfismo de árvores

$$count \quad = \quad (\![[\underline{0}\ , (\mathsf{succ} \cdot add \cdot \pi_2)]]\!) \tag{2}$$

onde $add\ (a, b) = a + b$. Conhecendo propriedades de length como, por exemplo,

$$\mathsf{length} \cdot cons \quad = \quad \mathsf{succ} \cdot \mathsf{length} \cdot \pi_2 \tag{3}$$

$$\mathsf{length} \cdot cat \quad = \quad add \cdot (\mathsf{length} \times \mathsf{length}) \tag{4}$$

use a lei de fusão-cata para completar a seguinte verificação de (1):

$$
\begin{array}{cl}
 & \mathsf{length} \cdot preord = count \\
\equiv & \qquad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
 & \mathsf{length} \cdot (\![[nil\ , (cons \cdot (id \times cat))]]\!) = (\![[\underline{0}\ , (\mathsf{succ} \cdot add \cdot \pi_2)]]\!) \\
\Leftarrow & \qquad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
 & \vdots
\end{array}
$$

RESPOSTA:
*(continue no verso se necessário, mas **não continue** no verso de outra página)* **Cot.:** ◯

Cálculo de Programas
2º Ano da LCC + LEI (Universidade do Minho)
Ano Lectivo de 2009/10
Avaliação Individual (Método A) TP4 —- Questão nr. 3

Identificação do Aluno:
Nome:
Número:

Sem Consulta(15 minutos)

Questão 4 - Relembre o tipo de dados

$$LTree\ \ a = Leaf\ \ a \mid Fork\ (LTree\ \ a, LTree\ \ a)$$

e as funções seguintes definidas como catamorfismos:

$listify :: LTree\ \ a \to [a]$
$count :: LTree\ \ a \to N_0$ - conta as folhas de uma $LTree\ \ a$.

Verifica-se que

$$length\ .\ listify = count \quad (1)$$

sendo *length* a função que conhece sobre listas.

Conhecendo propriedades do *lenght* como, por exemplo,

$$length\ .\ cat = add\ .\ (length \times length)$$

use a lei de fusão-cata para verificar a propriedade (1).

Cálculo de Programas

2.º Ano da LCC Universidade do Minho)
Ano Lectivo de 2009/10

Avaliação Individual (Método A) —- Questão nr. 4

IDENTIFICAÇÃO DO ALUNO:

Nome: . . . . . **Número** : | | | | | |

SEM CONSULTA (15 minutos)

Questão 1
Considere a seguinte declaração de *shape trees* (ie, árvores binárias sem dados)

```
data STree  = Tip | Fork (STree, STree)
```

e as seguintes funções (a primeira conta o número de folhas e a segunda "espelha" a árvore):

$$\begin{aligned} \mathsf{countLeaves}: &\quad STree \longrightarrow \mathbb{N} \\ \mathsf{countLeaves} = &\quad (\!|[\underline{1}, \mathsf{add}]|\!) \end{aligned}$$

$$\begin{aligned} \mathsf{mirror}: &\quad STree \longrightarrow STree \\ \mathsf{mirror} = &\quad (\!|[\mathsf{Tip}, \mathsf{Fork} \cdot \mathsf{swap}]|\!) \end{aligned}$$

Mostre, começando por aplicar a lei de fusão para os catamorfismos, que

$$\mathsf{countLeaves} \cdot \mathsf{mirror} \;=\; \mathsf{countLeaves}$$

Nota:
Recorde que a adição de naturais é comutativa, ie, $\mathsf{add} \cdot \mathsf{swap} = \mathsf{add}$ e que swap é um isomorfismo. Note ainda que $\mathsf{F}\, f = id + (f \times f)$.

RESPOSTA:
*(continue no verso se necessário, mas **não** continue no verso de outra página)* **Cot.:**

Nr. do aluno: | | | | | | – Folha 1 – Reservado aos docentes: | | |

Cálculo de Programas

2.º Ano da LCC e da LEI (Universidade do Minho)
Ano Lectivo de 2009/10

Turno TP-01: Avaliação Individual (Método A) —- Questão nr. 5

IDENTIFICAÇÃO DO ALUNO:

Nome: ..... **Número** : | | | | | |

SEM CONSULTA (15 minutos)

Questão 1 O apuramento do valor médio das folhas de uma árvore binária t

$$avg\ t = \frac{sum\ t}{count\ t}$$

de tipo *LTree* mostra a necessidade de fazer duas travessias de t, uma feita por $sum = (\!|[id\ ,\ add]|\!)$ e a outra por $count = (\!|[\underline{1}\ ,\ add]|\!)$, onde $add = \widehat{(+)}$. A lei de "banana-split"

$$\langle (\!|i|\!), (\!|j|\!)\rangle = (\!|h|\!) \quad \Leftarrow \quad h = \langle i \cdot \mathsf{F}\,\pi_1, j \cdot \mathsf{F}\,\pi_2\rangle \tag{1}$$

permite fazer o mesmo apuramento com uma só travessia:

$avg\ t = n\ /\ d$ **where**
  $(n, d) = aux\ t$
    $aux\ (Leaf\ a) = (a, 1)$
    $aux\ (Fork\ (x, y)) = (n1 + n2, d1 + d2)$
      where $(n1, d1) = aux\ (Fork\ x)$
        $(n2, d2) = aux\ (Fork\ y)$

Apresente justificações para os seguintes passos que já se deram no processo de conversão do par ("split") de catamorfimos sum e $count$ num único catamorfimo, de que o algoritmo acima deriva:

$$\begin{array}{ll}
 & \langle sum, count\rangle = (\!|h|\!) \\
\equiv & \quad \{\ \ldots\ \} \\
 & \langle (\!|[id\ ,\ add]|\!), (\!|[\underline{1}\ ,\ add]|\!)\rangle = (\!|h|\!) \\
\Leftarrow & \quad \{\ \ldots\ \} \\
 & h = \langle [id\ ,\ add] \cdot \mathsf{F}\,\pi_1, [\underline{1}\ ,\ add] \cdot \mathsf{F}\,\pi_2\rangle \\
\equiv & \quad \{\ \ldots\ \} \\
 & h = \langle [id\ ,\ add] \cdot (id + \pi_1 \times \pi_1), [\underline{1}\ ,\ add] \cdot (id + \pi_2 \times \pi_2)\rangle \\
\equiv & \quad \{\ \ldots\ \} \\
 & h = \langle [id\ ,\ (add \cdot (\pi_1 \times \pi_1))], [\underline{1}\ ,\ (add \cdot (\pi_2 \times \pi_2))]\rangle \\
\equiv & \quad \{\ \ldots\ \} \\
 & h = [\langle id, \underline{1}\rangle\ ,\ \langle (add \cdot (\pi_1 \times \pi_1)), (add \cdot (\pi_2 \times \pi_2))\rangle] \\
\equiv & \quad \{\ \ldots\ \} \\
 & h = [(\langle id, \underline{1}\rangle)\ , h2]\ \textbf{where}\ h2\ ((n1, d1), (n2, d2)) = (n1 + n2, d1 + d2)
\end{array}$$

RESPOSTA:
*(continue no verso se necessário, mas **não** continue no verso de outra página)*

Cot.:

Cálculo de Programas

2.º Ano da LCC e da LEI (Universidade do Minho)
Ano Lectivo de 2009/10

Turno TP-02: Avaliação Individual (Método A) —- Questão nr. 5

IDENTIFICAÇÃO DO ALUNO:

Nome: ..... **Número** : | | | | | |

SEM CONSULTA (15 minutos)

Questão 1 A lei de "banana-split"

$$\langle (\!|i|\!), (\!|j|\!) \rangle = (\!|h|\!) \quad \Leftarrow \quad h = \langle i \cdot \mathsf{F}\,\pi_1, j \cdot \mathsf{F}\,\pi_2 \rangle \tag{1}$$

permite-nos combinar um par de catamorfismos num só, reduzindo duas travessias de uma dada estrutura de dados (eg. lista, árvore, etc) a uma só.

Apresente justificações para o seguinte cálculo da lei (1):

$$\begin{array}{ll}
& \langle (\!|i|\!), (\!|j|\!) \rangle = (\!|h|\!) \\
\equiv & \quad \{ \quad \ldots\ldots \} \\
& \langle (\!|i|\!), (\!|j|\!) \rangle \cdot \mathbf{in} = h \cdot \mathsf{F}\langle (\!|i|\!), (\!|j|\!) \rangle \\
\equiv & \quad \{ \quad \ldots\ldots \} \\
& \langle ((\!|i|\!) \cdot \mathbf{in}\,), ((\!|j|\!) \cdot \mathbf{in}\,) \rangle = \langle h_1, h_2 \rangle \cdot \mathsf{F}\langle (\!|i|\!), (\!|j|\!) \rangle \\
\equiv & \quad \{ \quad \ldots\ldots \} \\
& \left\{ \begin{array}{l} (\!|i|\!) \cdot \mathbf{in} = h_1 \cdot \mathsf{F}\langle (\!|i|\!), (\!|j|\!) \rangle \\ (\!|j|\!) \cdot \mathbf{in} = h_2 \cdot \mathsf{F}\langle (\!|i|\!), (\!|j|\!) \rangle \end{array} \right. \\
\equiv & \quad \{ \quad \ldots\ldots \} \\
& \left\{ \begin{array}{l} i \cdot \mathsf{F}(\!|i|\!) = h_1 \cdot \mathsf{F}\langle (\!|i|\!), (\!|j|\!) \rangle \\ j \cdot \mathsf{F}(\!|j|\!) = h_2 \cdot \mathsf{F}\langle (\!|i|\!), (\!|j|\!) \rangle \end{array} \right. \\
\equiv & \quad \{ \quad \ldots\ldots \} \\
& \left\{ \begin{array}{l} i \cdot \mathsf{F}\pi_1 \cdot \mathsf{F}\langle (\!|i|\!), (\!|j|\!) \rangle = h_1 \cdot \mathsf{F}\langle (\!|i|\!), (\!|j|\!) \rangle \\ j \cdot \mathsf{F}\pi_2 \cdot \mathsf{F}\langle (\!|i|\!), (\!|j|\!) \rangle = h_2 \cdot \mathsf{F}\langle (\!|i|\!), (\!|j|\!) \rangle \end{array} \right. \\
\Leftarrow & \quad \{ \quad \ldots\ldots \} \\
& \left\{ \begin{array}{l} i \cdot \mathsf{F}\pi_1 = h_1 \\ j \cdot \mathsf{F}\pi_2 = h_2 \end{array} \right. \\
\equiv & \quad \{ \quad \ldots\ldots \} \\
& \langle i \cdot \mathsf{F}\pi_1, j \cdot \mathsf{F}\pi_2 \rangle = h
\end{array}$$

RESPOSTA:
*(continue no verso se necessário, mas **não** continue no verso de outra página)* **Cot.:**

Nr. do aluno: | | | | | | – Folha 1 – Reservado aos docentes: | | | |

Cálculo de Programas

2.º Ano da LCC e da LEI (Universidade do Minho)
Ano Lectivo de 2009/10

Turno TP-03: Avaliação Individual (Método A) —- Questão nr. 5

IDENTIFICAÇÃO DO ALUNO:

Nome: ..... **Número** : | | | | | |

SEM CONSULTA (15 minutos)

Questão 1 Pretendemos mostrar que a função

$$\begin{array}{l} bubble :: Ord\ a \Rightarrow [a] \rightarrow (a, [a]) \\ bubble\ [x] \quad = (x, [\,]) \\ bubble\ (x : l) = \mathbf{let}\ (y, m) = bubble\ l \\ \qquad\qquad\ \mathbf{in}\ \ \mathbf{if}\ x < y\ \mathbf{then}\ (x, y : m)\ \mathbf{else}\ (y, x : m) \end{array}$$

em que se baseia o algoritmo de "bubble sort" mais não é que a combinação de duas funções mutuamente recursivas: uma que calcula o mínimo de uma lista não vazia,

$$\begin{array}{l} getmin\ [x] = x \\ getmin\ (x : l) = min\ x\ (getmin\ l) \end{array}$$

e outra que remove esse mínimo da lista:

$$\begin{array}{l} remmin\ [x] = [\,] \\ remmin\ (x : l) = (max\ x\ (getmin\ l)) : (remmin\ l) \end{array}$$

Apresente justificações para o raciocínio que de seguida se faz no sentido de mostrar que, adoptando as seguintes definições e/ou abreviaturas:

$$\mathbf{in} = [singl\ ,\ cons] \tag{1}$$
$$lt\ (x, y) = x < y \tag{2}$$
$$mn = \widehat{min} = lt \rightarrow \pi_1, \pi_2 \tag{3}$$
$$mx = \widehat{max} = lt \rightarrow \pi_2, \pi_1 \tag{4}$$
$$aux = cons \cdot \langle (mx \cdot (id \times \pi_1)), (\pi_2 \cdot \pi_2) \rangle \tag{5}$$

se tem, fazendo $bubble = \langle getmin, remmin \rangle$:

$$\left\{ \begin{array}{l} getmin \cdot \mathbf{in} = [id\ ,\ mn \cdot (id \times getmin)] \\ remmin \cdot \mathbf{in} = [nil\ ,\ aux \cdot (id \times \langle getmin, remmin \rangle)] \end{array} \right.$$

$$\equiv \quad \left\{ \begin{array}{l} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \end{array} \right\}$$

$$\left\{ \begin{array}{l} getmin \cdot \mathbf{in} = [id\ ,\ mn \cdot (id \times \pi_1)] \cdot (id + id \times \langle getmin, remmin \rangle) \\ remmin \cdot \mathbf{in} = [nil\ ,\ aux] \cdot (id + id \times \langle getmin, remmin \rangle) \end{array} \right.$$

$$\equiv \quad \left\{ \begin{array}{l} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \end{array} \right\}$$

$$\langle getmin, remmin\rangle = (\!|\langle [id\ ,\ mn \cdot (id \times \pi_1)], [nil\ ,\ aux]\rangle|\!)$$

$$\equiv \left\{ \begin{array}{l} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \end{array} \right\}$$

$$bubble = (\!|[\langle id, nil\rangle\ ,\langle (mn \cdot (id \times \pi_1)), aux\rangle]|\!)$$

Justifique ainda o cálculo de parte do gene obtido:

$$mn \cdot (id \times \pi_1)$$

$$= \left\{ \begin{array}{l} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \end{array} \right\}$$

$$(lt \rightarrow \pi_1, \pi_2) \cdot (id \times \pi_1)$$

$$= \left\{ \begin{array}{l} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \end{array} \right\}$$

$$lt \cdot (id \times \pi_1) \rightarrow \pi_1 \cdot (id \times \pi_1), \pi_2 \cdot (id \times \pi_1)$$

$$= \left\{ \begin{array}{l} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \end{array} \right\}$$

$$lt \cdot (id \times \pi_1) \rightarrow \pi_1, \pi_1 \cdot \pi_2$$

Do mesmo modo se mostraria que

$$aux \quad = \quad lt \cdot (id \times \pi_1) \rightarrow cons \cdot \pi_2, cons \cdot (id \times \pi_2)$$

e logo (finalmente!)

$$\langle (mn \cdot (id \times \pi_1)), aux\rangle \quad = \quad lt \cdot (id \times \pi_1) \rightarrow \langle \pi_1, (cons \cdot \pi_2)\rangle, \langle (\pi_1 \cdot \pi_2), (cons \cdot (id \times \pi_2))\rangle$$

mas a justificação destes passos está fora do que se pede neste exercício.

RESPOSTA:
*(continue no verso se necessário, mas **não** continue no verso de outra página)*

Cot.:

Cálculo de Programas

2.º Ano da LCC e da LEI (Universidade do Minho)
Ano Lectivo de 2009/10

Turno TP-04: Avaliação Individual (Método A) —- Questão nr. 5

IDENTIFICAÇÃO DO ALUNO:

Nome: .... **Número** : [| | | |]

SEM CONSULTA (15 minutos)

Questão 1 Considere a seguinte função em Haskell

$$
\begin{array}{l}
mmap :: (a \rightarrow b) \rightarrow (a \rightarrow b) \rightarrow [a] \rightarrow ([b],[b]) \\
mmap\ f\ g = \langle f1, f2 \rangle\ \mathbf{where} \\
\quad f1\ [\,] = [\,] \\
\quad f1\ (a:l) = (f\ a) : f2\ l \\
\quad f2\ [\,] = [\,] \\
\quad f2\ (a:l) = (g\ a) : f1\ l
\end{array}
$$

que generaliza o map de listas a duas funções f e g, aplicadas alternadamente conforme se pode ver pelo regime de recursividade múltipla.

Pretendemos converter $mmap$ num catamorfismo de listas, usando para isso a lei que conhece. Justifique os passos que se deram já nesse processo de cálculo:

$$
\left\{
\begin{array}{l}
f1\ [\,] = [\,] \\
f1\ (a:l) = (f\ a) : f2\ l \\
f2\ [\,] = [\,] \\
f2\ (a:l) = (g\ a) : f1\ l
\end{array}
\right.
$$

$\equiv$ { .... }

$$
\left\{
\begin{array}{l}
f1 \cdot nil = nil \\
f1 \cdot cons = cons \cdot (f \times f2) \\
f2 \cdot nil = nil \\
f2 \cdot cons = cons \cdot (g \times f1)
\end{array}
\right.
$$

$\equiv$ { .... }

$$
\left\{
\begin{array}{l}
f1 \cdot [nil\ , cons] = [nil\ , (cons \cdot (f \times f2)] \\
f2 \cdot [nil\ , cons] = [nil\ , (cons \cdot (g \times f1)]
\end{array}
\right.
$$

$\equiv$ { .... }

$$
\left\{
\begin{array}{l}
f1 \cdot \mathbf{in} = [nil\ , (cons \cdot (f \times id))] \cdot (id + id \times f2) \\
f2 \cdot \mathbf{in} = [nil\ , (cons \cdot (g \times id))] \cdot (id + id \times f1)
\end{array}
\right.
$$

$\equiv$ { .... }

$$
\left\{
\begin{array}{l}
f1 \cdot \mathbf{in} = [nil\ , (cons \cdot (f \times id))] \cdot (id + id \times \pi_2) \cdot (id + id \times \langle f1, f2 \rangle) \\
f2 \cdot \mathbf{in} = [nil\ , (cons \cdot (g \times id))] \cdot (id + id \times \pi_1) \cdot (id + id \times \langle f1, f2 \rangle)
\end{array}
\right.
$$

$\equiv$ { .... }

$$
\left\{
\begin{array}{l}
f1 \cdot \mathbf{in} = [nil\ , (cons \cdot (f \times \pi_2))] \cdot (id + id \times \langle f1, f2 \rangle) \\
f2 \cdot \mathbf{in} = [nil\ , (cons \cdot (g \times \pi_1))] \cdot (id + id \times \langle f1, f2 \rangle)
\end{array}
\right.
$$

$$\equiv \quad \{ \quad \ldots\ldots\ldots\ldots \}$$

$$\langle f1, f2 \rangle \cdot \mathbf{in} = \langle [nil\ , (cons \cdot (f \times \pi_2))], [nil\ , (cons \cdot (g \times \pi_1))] \rangle \cdot (id + id \times \langle f1, f2 \rangle)$$

$$\equiv \quad \{ \quad \ldots\ldots\ldots\ldots \}$$

$$mmap\ f\ g \cdot \mathbf{in} = [\langle nil, nil \rangle\ , \langle (cons \cdot (f \times \pi_2)), (cons \cdot (g \times \pi_1)) \rangle] \cdot (id + id \times (mmap\ f\ g))$$

$$\equiv \quad \{ \quad \ldots\ldots\ldots\ldots \}$$

$$mmap\ f\ g = (\!|[\langle nil, nil \rangle\ , \langle (cons \cdot (f \times \pi_2)), (cons \cdot (g \times \pi_1)) \rangle]|\!)$$

RESPOSTA:
*(continue no verso se necessário, mas **não** continue no verso de outra página)*

Cot.:

Cálculo de Programas

2.º Ano da LCC e da LEI (Universidade do Minho)
Ano Lectivo de 2009/10

Turno TP-LCC: Avaliação Individual (Método A) —- Questão nr. 5

IDENTIFICAÇÃO DO ALUNO:

Nome: ..... **Número** : | | | | | |

SEM CONSULTA (15 minutos)

Questão 1 A lei de "banana-split"

$$\langle (\!|i|\!), (\!|j|\!) \rangle = (\!|h|\!) \quad \Leftarrow \quad h = \langle i \cdot \mathsf{F}\,\pi_1, j \cdot \mathsf{F}\,\pi_2 \rangle \tag{1}$$

permite-nos combinar um par de catamorfismos num só, reduzindo duas travessias de uma dada estrutura de dados (eg. lista, árvore, etc) a uma só.

Apresente justificações para o seguinte cálculo da lei (1):

$$\begin{array}{ll}
 & \langle (\!|i|\!), (\!|j|\!) \rangle = (\!|h|\!) \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
 & \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
 & \langle (\!|i|\!), (\!|j|\!) \rangle \cdot \mathbf{in} = h \cdot \mathsf{F}\langle (\!|i|\!), (\!|j|\!) \rangle \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
 & \langle ((\!|i|\!) \cdot \mathbf{in}\,), ((\!|j|\!) \cdot \mathbf{in}\,) \rangle = \langle h_1, h_2 \rangle \cdot \mathsf{F}\langle (\!|i|\!), (\!|j|\!) \rangle \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
 & \left\{ \begin{array}{l} (\!|i|\!) \cdot \mathbf{in} = h_1 \cdot \mathsf{F}\langle (\!|i|\!), (\!|j|\!) \rangle \\ (\!|j|\!) \cdot \mathbf{in} = h_2 \cdot \mathsf{F}\langle (\!|i|\!), (\!|j|\!) \rangle \end{array} \right. \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
 & \left\{ \begin{array}{l} i \cdot \mathsf{F}(\!|i|\!) = h_1 \cdot \mathsf{F}\langle (\!|i|\!), (\!|j|\!) \rangle \\ j \cdot \mathsf{F}(\!|j|\!) = h_2 \cdot \mathsf{F}\langle (\!|i|\!), (\!|j|\!) \rangle \end{array} \right. \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
 & \left\{ \begin{array}{l} i \cdot \mathsf{F}\pi_1 \cdot \mathsf{F}\langle (\!|i|\!), (\!|j|\!) \rangle = h_1 \cdot \mathsf{F}\langle (\!|i|\!), (\!|j|\!) \rangle \\ j \cdot \mathsf{F}\pi_2 \cdot \mathsf{F}\langle (\!|i|\!), (\!|j|\!) \rangle = h_2 \cdot \mathsf{F}\langle (\!|i|\!), (\!|j|\!) \rangle \end{array} \right. \\
\Leftarrow & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
 & \left\{ \begin{array}{l} i \cdot \mathsf{F}\pi_1 = h_1 \\ j \cdot \mathsf{F}\pi_2 = h_2 \end{array} \right. \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
 & \langle i \cdot \mathsf{F}\pi_1, j \cdot \mathsf{F}\pi_2 \rangle = h
\end{array}$$

RESPOSTA:
*(continue no verso se necessário, mas **não** continue no verso de outra página)* **Cot.:**

Nr. do aluno: | | | | | | – Folha 1 – Reservado aos docentes: | | | |

Cálculo de Programas
2º Ano da LCC + LEI (Universidade do Minho)
Ano Lectivo de 2009/10
Avaliação Individual (Método A) TP1 —- Questão nr. 6

Identificação do Aluno:
Nome:
Número:

Sem Consulta(15 minutos)

Questão 6 - Use a lei da absorção-cata, entre outras, para verificar a seguinte propriedade sobre listas:

$$(expo\ x)\ .\ product\ =\ product\ .\ map\ (expo\ x)$$

sendo
```
expo x y = y^x ,
mult (x,y) = x*y
```
e
$product = (\!|[1, mult]|\!)$.

Cálculo de Programas
2º Ano da LCC + LEI (Universidade do Minho)
Ano Lectivo de 2009/10
Avaliação Individual (Método A) TP2 —- Questão nr. 6

Identificação do Aluno:
Nome:
Número:

Sem Consulta(15 minutos)

Questão 6 - Use a lei da absorção-cata, entre outras, para verificar a seguinte propriedade sobre listas:

$$length \;=\; sum \;.\; map\,\underline{1}$$

sendo *lenth*, *sum* e *map* as funções que conhece sobre listas.

Cálculo de Programas

2.º Ano da LCC e da LEI (Universidade do Minho)
Ano Lectivo de 2009/10

Turno TP-03: Avaliação Individual (Método A) —- Questão nr. 6

IDENTIFICAÇÃO DO ALUNO:

Nome: . . . . . **Número** : | | | | | |

SEM CONSULTA (15 minutos)

Questão 1 Pretendemos mostrar que a propriedade distributiva da multiplicação em relação à adição

$$a * \sum_{i=1}^{n} b_i \quad = \quad \sum_{i=1}^{n} (a * b_i) \tag{1}$$

se generaliza a árvores, por exemplo, do tipo LTree,

$$(a*) \cdot \mathsf{sum} \quad = \quad \mathsf{sum} \cdot (\mathsf{LTree}\ (a*)) \tag{2}$$

em que, como sabe

$$\mathsf{LTree}\ f = (\!|\mathbf{in} \cdot (f + id)|\!) \tag{3}$$
$$\mathsf{sum} = (\!|[id\ , \widehat{(+)}]|\!) \tag{4}$$
$$\mathbf{in} = [Leaf\ , Fork] \tag{5}$$
$$\mathsf{F}\ f = id + f^2 \tag{6}$$

Demonstre a propriedade (2) recorrendo às leis de *fusão-* e *absorção-*cata e a propriedades elementares da aritmética como, por exemplo, $a * (b + c) = a * b + a * c$ que, com variáveis, diz o mesmo que

$$(a*) \cdot \widehat{(+)} = \widehat{(+)} \cdot ((a*) \times (a*)) \tag{7}$$

RESPOSTA:
*(continue no verso se necessário, mas **não** continue no verso de outra página)* **Cot.:**

Nr. do aluno: | | | | | | – Folha 1 – Reservado aos docentes: | | | |

Cálculo de Programas
2º Ano da LCC + LEI (Universidade do Minho)
Ano Lectivo de 2009/10
Avaliação Individual (Método A) TP4 —- Questão nr. 6

Identificação do Aluno:
Nome:
Número:

Sem Consulta(15 minutos)

Questão 6 - Relembre o tipo de árvores binárias:

```
data LTree a = Leaf a | Fork (LTree a, LTree a)
```

Use a lei da absorção-cata, entre outras, para verificar a seguinte propriedade sobre "leaf trees":

$$count_{LT} \cdot mirror = count_{LT} \cdot map_{LT}\ f$$

sendo $count_{LT} = (\!|[\underline{1}, add]|\!)$ e $mirror = (\!|in \cdot (id + swap)|\!)$.

Cálculo de Programas

2.º Ano da LCC Universidade do Minho)
Ano Lectivo de 2009/10

Avaliação Individual (Método A) —- Questão nr. 6

IDENTIFICAÇÃO DO ALUNO:

Nome: . . . . . **Número** : | | | | | |

SEM CONSULTA (15 minutos)

Questão 1
Considere a seguinte declaração de *árvores binárias*

```
data BTree a = Empty | Node (a, (BTree a, BTree a))
```

e as seguintes funções (a primeira conta o número de nodos na árvore e a segunda soma os valores armazenados nos seus nodos):

$$\begin{aligned} \mathsf{countBTree}: &\quad BTree\, A \longrightarrow I\!N \\ \mathsf{countBTree} = &\quad (\!|[\underline{0}, \mathsf{add} \cdot (\underline{1} \times \mathsf{add})]|\!) \end{aligned}$$

$$\begin{aligned} \mathsf{sumBTree}: &\quad BTree\, I\!N \longrightarrow BTree\, I\!N \\ \mathsf{sumBTree} = &\quad (\!|[\underline{0}, \mathsf{add} \cdot (id \times \mathsf{add})]|\!) \end{aligned}$$

Mostre, começando por aplicar a lei de absorção para os catamorfismos, que

$$\mathsf{sumBTree} \cdot (\mathsf{mapBTree}\, \underline{1}) \;=\; \mathsf{countBTree}$$

Nota:
Recorde que $\mathsf{mapBTree}\, f = (\!|\mathsf{in} \cdot B(f, id)|\!)$ onde $\mathsf{in} = [\mathsf{Empty}, \mathsf{Node}]$ e $B(f, g) = id + f \times (g \times g)$.

RESPOSTA:
*(continue no verso se necessário, mas **não** continue no verso de outra página)* **Cot.:**

Nr. do aluno: | | | | | | – Folha 1 – Reservado aos docentes: | | |