Métodos Formais de Programação I +
Opção I - Métodos Formais de Programação I

4.º Ano da LMCC (7007N2) + LESI (5307P6)
Ano Lectivo de 2003/04

Exame (época especial) — 16 de Setembro 2004
14h00
Sala 1211

NB: *Esta prova consta de **7** alíneas todas com as mesma cotação.*

PROVA SEM CONSULTA (2 horas)

Questão 1 Uma das operações mais básicas de um pacote de 'data mining' é aquela que permite aglutinar ou sumariar dados numéricos de acordo com critérios vários de classificação. Por exemplo, sejam dados três quadros,

Mês	**Total (Euro $\times 10^6$)**
Janeiro	21
Fevereiro	10
Março	9
Abril	15
Maio	9
Junho	9
Julho	10
Agosto	12
Setembro	7
Outubro	5
Novembro	3
Dezembro	30

(a) — Vendas

Mês	**Estação**
Janeiro	Inverno
Fevereiro	Inverno
Março	Primavera
Abril	Primavera
Maio	Primavera
Junho	Verão
Julho	Verão
Agosto	Verão
Setembro	Outono
Outubro	Outono
Novembro	Outono
Dezembro	Inverno

(b) — Critério "Estação"

Mês	**Vendedor**
Janeiro	A
Fevereiro	A
Março	B
Abril	B
Maio	C
Junho	C
Julho	A
Agosto	A
Setembro	B
Outubro	B
Novembro	C
Dezembro	C

(c) — Critério "Vendedor"

em que: (a) corresponde à relação de vendas, por mês, de uma dada firma comercial; (b) corresponde ao critério que classifica meses em estações do ano; e (c) corresponde ao critério que relaciona meses com os vendedores da firma.

As duas tabelas que se seguem correspondem à aglutinação de (a) seguindo, respectivamente, o critério (b) e o critério (c):

Estação	**Total (Euro $\times 10^6$)**
Inverno	61
Primavera	33
Verão	31
Outono	15

(d) — Critério "Estação"

Vendedor	**Total (Euro $\times 10^6$)**
A	53
B	36
C	51

(e) — Critério "Vendedor"

Repare que as tabelas (a), (d) e (e) são multiconjuntos de tipo genérico

```
map @A to nat1
```

e as tabelas (b) e (c) são funções de classificação de tipo genérico

```
map @A to @B
```

1. Especifique em VDM-SL a função

   ```
   msetTot: (map @A to nat1) -> nat
   ```

   que calcula o total das multiplicidades de um multiconjunto.

2. Especifique em VDM-SL a função

   ```
   msetAgl: (map @A to nat1) * (map @A to @B) -> (map @B to nat1)
   ```

que realiza a operação de aglutinação que acima se ilustrou.

Questão 2 Sejam dadas suas funções f, g tais que $g \cdot f \quad = \quad g$, e f é diferente de id. Será este facto suficiente para dizermos que g **não** é injectiva? Justifique.

Questão 3 Sejam $\underline{a}$ e $\underline{b}$ duas funções constantes. Qual o tipo e o significado da relação $\underline{a} \cdot \underline{b}^{\circ}$? Responda introduzindo variáveis e simplificando.

Questão 4 Mostre que a definição relacional do condicional de McCarthy,

$$P \rightarrow R\ ,\ S \quad = \quad (R \cdot dom\ P) \cup (S \cdot \neg(dom\ P)) \tag{1}$$

(onde, para X correflexiva, $\neg X$ abrevia $id - X$) decorre da propriedade universal:

$$P \rightarrow R\ ,\ S \quad \subseteq \quad Y \quad \equiv \quad R \cdot dom\ P \subseteq Y \quad \wedge \quad S \cdot \neg(dom\ P) \subseteq Y \tag{2}$$

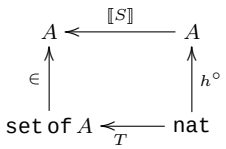
Questão 5 Dos seguintes tipos de dados, escritos em notação VDM-SL, um é seu conhecido das aulas laboratoriais desta disciplina,

```
FamilyOfSets = map nat to set of A;
HTable       = FamilyOfSets
               inv t == forall n in set dom t &
                        forall a in set t(n) & h(a) = n;
A = token;
```

em que `h : A -> nat` é uma função de "hashing" predefinida.

1. Que nem toda a `FamilyOfSets` é uma `HTable` é fácil de ver, por causa do invariante desta última: pode haver elementos armazenados no índice errado. Especifique uma função, `filter : (A -> nat) -> FamilyOfSets -> HTable` , que extraia de uma `FamilyOfSets` x a "maior" tabela de *hashing* que x contém, para uma dada função de *hashing*.
2. Considere a expressão e diagrama que se seguem,

$$\in \cdot T \quad = \quad [\![S]\!] \cdot h^{\circ} \tag{3}$$



onde T é uma tabela de *hashing* (relação simples) e S é o conjunto de elementos de A que a tabela armazena.

Converta (3) no correspondente predicado em VDM-SL e use-o para definir a **pós-condição** de uma operação que relaciona um dado S (de tipo `set of A`) com a tabela de *hashing* T (de tipo `HTable`) que o armazena.
