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

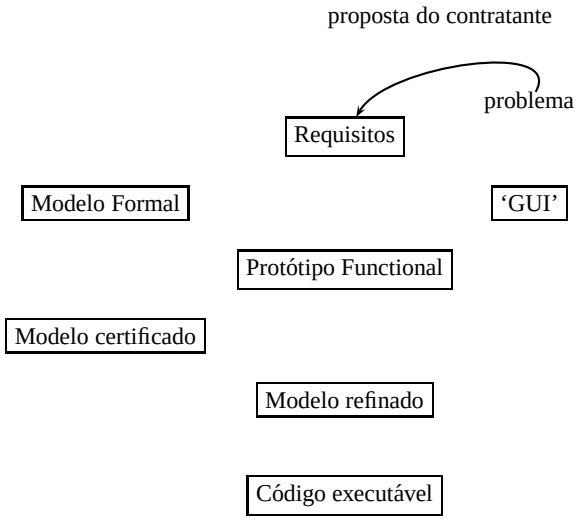
4.º Ano de LMCC (7007N2) + LESI (5307P6)
Ano Lectivo de 2006/07

Exame (época de recurso) — 7 de Fevereiro 2008
14h00
Sala 1213

NB*: Esta prova consta de **8** alíneas todas com as mesma cotação.*

PROVA SEM CONSULTA (2 horas)

Questão 1 O método adoptado nesta disciplina para desenvolvimento de *'software'* segue o chamado *ciclo de vida de Balzer*, ciclo esse que envolve a produção dos documentos que se apresentam no diagrama que se segue:



1. Complete o diagrama com setas indicativas do fluxo da informação e articulação entre as várias fases do processo.
2. Estabeleça as diferenças entre linguagem de *programação*, linguagem *natural* e linguagem de *especificação formal* e indique em que documentos (fases) do referido ciclo de vida são empregues que tipos de linguagens e porquê.

Questão 2 Considere os requisitos de uma instituição académica que se seguem,

> *Em cada ano lectivo, cada aluno pode frequentar entre uma e 6 disciplinas (todas opcionais), sendo 25 o número máximo de alunos por disciplina. Considera-se que um aluno está a frequentar uma disciplina sempre que a sua nota é indeterminada, F(altou), R(eprovado) ou D(esistiu). Uma nota de 10 a 20, inclusivé, indica que o aluno já realizou a disciplina.*

para os quais se construiu o modelo VDM-SL que se segue:

- Classificações:

```
Classif = <I> | <F> |<R> | <D> | Positiva ;
Positiva = nat
   inv n == n in set {10,...,20};
```

- Relação alunos-disciplinas (onde `AId` e `DId` são tokens):

```
Classifs = map (AId * DId) to Classif
   inv R == let I = { x | x in set dom R & R(x) in set {<I>,<F>,<R>,<D>}}
            in forall mk_(a,d) in set I &
                   let x = card {d' | mk_(a',d') in set I & a = a'},
                       y = card {a' | mk_(a',d') in set I & d = d'}
                   in  y <= 25 and x >= 1 and x <= 6;
```

- Estado do sistema:

```
State  :: alunos:      map AId to Nome
          disciplinas: map DId to Nome
          classifs :   Classifs
       inv mk_State(M,N,R) ==
             {d | mk_(-,d) in set dom R} subset dom N  and
             {a | mk_(a,-) in set dom R} subset dom M;
```

onde `Nome` é token.

1. Sabendo que um plano de estudos é um conjunto de disciplinas,

   ```
   PE = set of DId
   ```

   especifique a operação que verifica se um aluno já completou um dado plano de estudos:

   ```
   done : AId * PE * State -> bool
   done(a,p,db) == ..................
   ```

2. Especifique agora a operação de inscrição de um aluno a uma dada disciplina,

   ```
   Inscr : AId * DId * State -> State
   Inscr(a,d,db) == ..........................................
   pre ........................................................
   ```

   prestando especial atenção à pre-condição necessária para que os invariantes em jogo sejam preservados.

3. Formule (em notação VDM-SL) a obrigação de prova associada à operação `Inscr`. Por que razão não é necessária uma obrigação de prova semelhante para a função `done`? Justifique.

Questão 3 Qual é a condição necessária e suficiente para que, em VDM-SL, a expressão $\{a \mapsto b\} \cup \{c \mapsto d\}$ designe um *mapping* válido? Baseando-se, entre outras, nas propriedades

$$\lceil \{x \mapsto y\} \rceil = \underline{y} \cdot \underline{x}^{\circ} \quad (1)$$

$$M \cup N \text{ simples} \equiv M \text{ simples} \wedge N \text{ simples} \wedge M \cdot N^{\circ} \subseteq id \quad (2)$$

(onde $\underline{k}$ designa a função cujo resultado é sempre k) complete a justificação que se segue do cálculo (via transformada-PF) dessa condição:

$\lceil \{a \mapsto b\} \cup \{c \mapsto d\} \rceil$ simples

$\equiv$ { ..... }

$\underline{b} \cdot \underline{a}^{\circ} \cup \underline{d} \cdot \underline{c}^{\circ}$ simples

$\equiv$ { ..... }

$$\underline{b} \cdot \underline{a}^\circ \cdot \underline{c} \cdot \underline{d}^\circ \subseteq id$$
$$\equiv \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \}$$
$$\underline{a}^\circ \cdot \underline{c} \subseteq \underline{b}^\circ \cdot d$$
$$\equiv \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \}$$
$$a = c \Rightarrow b = d$$

Questão 4 Complete o seguinte raciocínio que mostra que, se $\leq$ for uma ordem *antissimétrica* e *reflexiva*, então a implicação

$$f\ a = f\ b \Rightarrow a \leq b \qquad (3)$$

é *equivalente* à afirmação de f como função injectiva:

f é injectiva
$$\equiv \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \}$$
$$f^\circ \cdot f \subseteq id$$
$$\equiv \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \}$$
$$f^\circ \cdot f \subseteq \leq \cap \leq^\circ$$
$$\equiv \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \}$$
$$f^\circ \cdot f \subseteq \leq \ \wedge\ f^\circ \cdot f \subseteq \leq^\circ$$
$$\equiv \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \}$$
$$f^\circ \cdot f \subseteq \leq$$
$$\equiv \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \}$$
(3)

Questão 5 O manual "on-line" de VDM-SL fornece a seguinte informação sobre um operador da linguagem:

Operator	Name	Type	Semantics description
`inverse m`	Map inverse	inmap A to $B \rightarrow$ inmap B to A	yields the inverse map of `m`. `m` must be a 1-to-1 mapping.

Recordando que a semântica de um 'mapping' em VDM-SL é o de uma relação simples e que o símbolo `inmap` identifica mappings injectivos, justifique a pré-condição que consta da descrição semântica de `inverse m` acima com base num quadro que conhece bem:

	Reflexive	*Coreflexive*
ker R	entire R	injective R
img R	surjective R	simple R

(4)
