Especificação e Desenvolvimento Formal de 'Software'

Mestrado em Informática + Curso de Especialização em Informática
Ano Lectivo de 1999/2000

Exame (Época normal) — 11 de Fevereiro de 2000
9h30
Sala do Mestrado

NB: *Esta prova consta de **8** questões de **2.5 valores** cada uma.*

PROVA COM CONSULTA $(2\ horas)$

Questão 1 Considere o seguinte fragmento de VDM-SL:

```
types

People = map token to Person;

Person :: name       : seq1 of char
          age        : nat1
          ifStudent  : [ Student ]
          ifEmployed : [ Employed ];

Student :: school : School
           number : nat1;

Employed :: company : Company
            job     : JobDescriptor;

School = token;

Company = token;

JobDescriptor = token;
```

Converta este fragmento para VDM++ e desenhe o correspondente diagrama de classes em UML.

Questão 2 Recorde o Exercício 13 que foi abordado nas aulas desta disciplina (pp.22–25 de VDM++ LECTURE NOTES EXERCISES). Comente as diferenças entre a especificação apresentada nesse exercício e a que se segue:

```
types

Point :: x : nat
         y : nat;

StoreName = token;

Object :: xlength : nat
          ylength : nat;

Store :: contents : map Point to Object
         xbound   : nat
         ybound   : nat
```

```
        inv mk_Store(contents,xbound,ybound) == ... ;

Site = map StoreName to Store;
```

Qual das duas especificações escolheria? **NB:** omite-se o invariante sobre `Store` que é uma adaptação óbvia daquele que é fornecido na especificação original.

Questão 3 Recorde do 'slide' E-34 que a definição explícita do algoritmo `MergeSort` depende da definição (implícita ou explícita) de `Merge`. Indique qual das seguintes definições alternativas para `Merge` escolheria e porquê:

1. ```
   Merge (l: seq of real, r: seq of real) o: seq of real
   post IsPermutation(o,l^r) and IsOrdered(o) ;
   ```
2. ```
   Merge: seq of real * seq of real -> seq of real
   Merge(l,r) == l ^ r ;
   ```
3. ```
   Merge: seq of real * seq of real -> seq of real
   Merge(l,r) == if      l=[] then r
                 else if r=[] then l
                 else let x = hd(l), xs= tl(l),
                          y = hd(r), ys= tl(r)
                      in if x < y
                         then [x] ^ merge(xs,[y]^ys)
                         else [y] ^ merge([x]^xs,ys);
   ```
4. ```
   Merge (l: seq of real, r: seq of real) o: seq of real
   pre IsOrdered(l) and IsOrdered(r)
   post IsPermutation(o,l^r) and IsOrdered(o) ;
   ```

NB: não se esqueça de justificar a sua não-escolha das outras, escalonando-as quando aos seus méritos ou defeitos.

Questão 4 Adapte a especificação de `MergeSort` dada nesse slide à de `Quicksort`, isto é, complete as reticências em

```
QuickSort: seq of real -> seq of real
QuickSort(l) == cases l :
                [ ]     -> l,
                [h] ^ t -> let l1 = ...,
                               l2 = ...
                           in let l_1 = QuickSort(l1),
                                  l_2 = QuickSort(l2)
                              in l_1 ^ ..... ^ l_2
                end;
```

Questão 5 Indique qual é o erro que obterá na 'toolbox' de VDM-SL se tentar calcular a expressão

```
{ i * i |-> i  |  i in set {-1,...,2}}
```

Com base neste exemplo, considere a expressão

```
{ f(i) |-> g(i) | i in set s }
```

onde `f` e `g` são funções arbitrárias e `s` é um subconjunto da intersecção dos domínios de `f` e de `g`. Indique em quais das situações que abaixo se descrevem essa expressão tem problemas de definição e porquê:

1. O conjunto `s` tem no máximo 1 elemento.
2. A função `f` é constante, isto é `f(i) == k` qualquer que seja `i`.
3. O facto `forall i,j in set s & i <> j => f(i) <> f(j)` verifica-se.

Questão 6 Indique através de contra-exemplos quais das seguintes igualdades não são válidas em VDM-SL/VDM++, onde `f` é uma função parcial finita arbitrária:

1. `dom f <-: f = f`
2. `{} <: f = {|->}`
3. `dom f <: f = f`
4. `f munion f = f`
5. `f = f ++ f`

Questão 7 Aquando da passagem do ano de 1999 para 2000 o mundo inteiro foi sensível a um "tabu informático" ignorado durante décadas: o problema Y2K. Mais tarde muita gente veio a questionar-se sobre a real importância do problema. Ao fim e ao cabo, poucos abordaram o mais importante: que há um problema que é de facto insolúvel e só pode ser resolvido a prazo, já que — em rigor — datas são números reais (dízimas infinitas) que apenas podem ser "aproximadas" por quantidades finitas como dias, meses e anos.

Para se apreciar os problemas tipo Y2K do passado, leia-se o seguinte excerpto de um "Lunário Perpétuo" do séc. XVIII [1]:

> "**Do Ano e Sua Divisão** — (...) Júlio César instituiu o ano, de que hoje usamos, de 365 dias e 6 horas, a qual quantidade não é exacta, pois vemos claramente adiantar-se o tempo; (...) a Santa Madre Igreja usa do ano que instituiu Júlio César, tomando em cada ano as 6 horas, que formam um dia inteiro em cada quatro anos, chamando-se bissexto a esse ano, a que se acrescenta um dia (...).
>
> **Da Reforma do Calendário** — Tendo-se observado, que desde a celebração do concílio de Niceia, em 325, até ao ano de 1582, se haviam antecipado os equinócios 10 dias do assento fixo em que os colocara Dionísio Romano; (...) mandou o papa Gregório XIII proceder à reforma do Calendário, em virtude da qual se determinou: 1.º que no mês de Outubro de 1582 se suprimissem 10 dias, contando 4 no dia de S.Francisco, e 15 no seguinte; 2.º que em cada 400 anos se suprimissem 3 dias, principiando de 1700, 1800, 1900, 2100, 2200, 2300, 2500, *etc.* (que por isso não são bissextos), para diminuir o excesso do ano sinodal ao civil, e os equinócios ficarem imóveis a 21 de Março e 23 de Setembro (...)."

O que se segue é um esboço de um tipo em VDM-SL para a noção de data, cujo invariante tenta especificar — mas erradamente — as irregularidades que o Lunário descreve:

```
types

Date :: year: nat
        month: nat1
        day: nat1
     inv mk_Date(y,m,d) ==
         if m in set {1,3,5,7,8,12}  then d <= 31
         else if m in set {4,6,9,11} then d <= 30
         else if m=10 and y=1582 then d<5 or d>14
         else if m=2 and leapYear(y) then d <= 29
         else if m=2 and not leapYear(y) then d <= 28
         else false;
```

onde

```
leapYear(y) : nat -> Bool
leapYear(y) == 0 = rem(y, if 1700 < y or rem(y,100)=0 then 400 else 4)
```

Quais são os problemas que afectam a definição deste invariante? Justifique.

Questão 8 Enumere sumariamente vantagens e as desvantagens dos métodos ensinados nesta disciplina e faça um balanço (negativo/positivo) dessa sua enumeração.

---

[1] In *Lunário de Prognóstico Perpétuo, para Todos os Reinos e Províncias*, por Jerónimo Cortez, Valenciano, re-edição Lello & Irmão, 1910.