Paradigmas da Programação I

ESI (5301P3) / MCC (7001N4)

Ano Lectivo de 2001/2002

Questão 1 Ao consultar a página da Santa Casa da Misericórdia pode obter-se os resultados do totoloto sob a forma de uma lista de pares `(Chave,Suplementar)`, em que `Chave` é composto por seis números (por ordem crescente) e `Suplementar` é um número (todos estes números são inteiros entre 1 e 49). Para facilitar todo o processamento, vamos admitir que a `Chave` é uma lista.

```
type Resultado = (Chave,Suplementar)
type Chave = [Numero]
type Suplementar = Numero
type Numero = Int
```

1. Defina uma função que teste se um dado `Resultado` é válido (i.e., que a `Chave` está ordenada por ordem crescente, tem seis elementos e não tem repetições e que o `Suplementar` não pertence à `Chave`).

2. Defina agora uma função que, dadas duas `Chaves` (satisfazendo as restrições acima) calcule o número de `Numeros` comuns.

3. O segundo prémio do totoloto corresponde a acertar em cinco dos seis números da `Chave` e no número `Suplementar`. Defina uma função que teste se dada uma `Chave` e um `Resultado`, essa chave corresponde ou não ao segundo prémio.

4. Defina uma função que dada uma `Chave` e um `Resultado` dê como resultado o prémio obtido por essa `Chave` – um número de 0 a 5 em que 0 corresponde a não haver prémio e cada um dos outros resultados corresponde ao prémio obtido (1 corresponde ao primeiro, 2 ao segundo ...).

Questão 2 Pretende-se representar em Haskell estruturas de dados *Grafo Orientado*. Um grafo é constituído por um conjunto de nós e um conjunto de setas com origem e destino em nós (não podendo haver mais do que uma seta com a mesma origem e destino). Considere as seguintes definições de tipos:

```
type No = Int
type Seta = (No,No)
type Grafo = [Seta]
```

em que um par $(2,3)$ representa uma seta com origem no nó 2 e destino no nó 3.

1. Escreva uma função `uniao :: Grafo -> Grafo -> Grafo` que calcule a união de dois grafos, isto é, um grafo contendo todas as as setas de ambos os grafos que recebe como argumentos. Observe que uma seta que exista em ambos os grafos deverá ocorrer apenas uma vez no grafo resultante da unio.

2. Considere a seguinte definição alternativa de tipos para a representação de grafos:

   ```
   type OutroGrafo = [(No,[No])]
   ```

   Nesta representação associa-se a cada nó x o conjunto de nós que são destino de setas com origem em x. Por exemplo, as representações `[(1,2),(1,3),(2,3)] :: Grafo` e `[(1,[2,3]), (2,[3]), (3,[])] :: OutroGrafo` referem-se ao mesmo grafo.

   Escreva uma função de tipo `OutroGrafo -> Grafo` de conversão da segunda representação na primeira.

Questão 3 Considere o seguinte registo correspondente a uma submissão electrónica dum trabalho prático no formato XML especificado para o efeito:

Nome:_____ **Número:**_____ **Curso:**_____

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<submisso>
  <equipe>
    <aluno>
      <numero>4238</numero>
      <nome>Jos Alberto Rodrigues</nome>
      <curso>LESI</curso>
    </aluno>
    <aluno>
      <numero>4140</numero>
      <nome>Jos Carlos Ramalho</nome>
      <curso>LESI</curso>
    </aluno>
    <aluno>
      <numero>4156</numero>
      <nome>Paulo Jorge Domingues</nome>
      <curso>LESI</curso>
    </aluno>
  </equipe>
  <fich-env>ZXcompiler.zip</fich-env>
  <data-hora>1986-03-26T16:05:13</data-hora>
</submisso>
```

Como pode ver, uma submissao é composta por uma equipe, um ficheiro enviado fich-env e uma hora data-hora. Por sua vez, equipe é composta por uma sequência de alunos em que cada aluno é composto por numero, nome e curso. Responda sucintamente a cada uma das alíneas seguintes:

1. Defina em Haskell os tipos de dados abstractos Submissao, Submissoes e Aluno.
2. Defina uma instância da classe Show para cada um destes tipos (o formato de saída fica ao seu critério).
3. Defina uma função que recebendo um elemento de Submissoes produz uma lista de alunos, ordenados alfabeticamente por nome.
4. Um invariante engraçado que todas as instâncias de Submissoes deverão satisfazer é a inexistência de alunos repetidos em submissões diferentes. Defina uma função que dada uma instância de Submissoes produz uma lista com os alunos repetidos (utilize o numero para procurar e comparar os alunos).

Nome:_____ **Número:**_____ **Curso:**_____