Programação Funcional

Teste 317

25 de Março de 2017

DCC/FCUP

Nome: _____

Nº mecanográfico: _____

- **Este teste contém 7 questões e 3 páginas.**
- **Responda às questões no espaço marcado no enunciado.**
- **Se nada for dito explicitamente em contrário, pode usar funções auxiliares e/ou do prelúdio-padrão de Haskell**

1. (30%) Responda a cada uma das seguintes questões, indicando **apenas** o resultado de cada expressão.

(a) `1:(5:(4:(3:[])))` = _____

(b) `tail [4,5,6,9]` = _____

(c) `head ([2,3] ++ [1,4] ++ [4,6])` = _____

(d) `drop 5 [0,3..30]` = _____

(e) `length ([1,2]:[]:[3,4]:[[5]])` = _____

(f) `[ x*y | x <- [1..3], y <- [x..3]]` = _____

(g) `[ x | x <- [1..3], y <- [1..3], (x+y) == 4]` = _____

(h) Sem usar explicitamente a lista dada, defina a seguinte lista em compreensão:

`[0,-1,2,-3,4,-5,6,-7,8,-9,10]` = _____

(i) Considere a seguinte definição em Haskell:

```
h [] = 0
h (x:xs) = 1 + h xs
```

A avaliação da expressão `h [0..7]` tem como resultado: _____

(j) Indique um tipo admissível para `(['1','2','3'],[1.0,2.0,3.0])`:

_____

(k) Indique o tipo mais geral da função:

```
fst (x,y) = x
```

_____

(l) Considere a seguinte função em Haskell:

```
h x y z = x >= z && y <= z
```

Indique um tipo admissível para a função `h`:

_____

(m) Indique o tipo mais geral da função `f` definida como `f xs = xs!!2`:

_____

2. (10%) Escreva uma função `numEqual` tal que `numEqual n m p` é o número de `n`, `m` e `p` que são iguais. Por exemplo, `numEqual 1 1 1 = 3` e `numEqual 7 5 5 = 2`.

3. (10%) A área de um quadrilátero é dada pela seguinte fórmula:

$$area = \frac{1}{4}\sqrt{4p^2q^2 - (b^2 + d^2 - a^2 - c^2)^2}$$

onde a, b, c e d são as medidas dos lados e p e q são as medidas das diagonais. Defina uma função `area`, tal the `area a b c d p q` calcula a área de um quadrilátero.

4. (10%) Defina recursivamente a função `enquantoPar`, tal que `enquantoPar xs` retorna o maior prefixo da lista `xs` em que todos os elementos são pares. Por exemplo, `enquantoPar [2,4,8,3,4,8,6] = [2,4,8]`.

5. (10%) Considere a função `nat_zip` que dada uma lista `xs`, devolve uma lista em que cada elemento é o par constituído pelo número que indica a posição de cada elemento na lista e pelo elemento que ocupava essa posição. Por exemplo: `nat_zip "zip"= [(1, 'z'), (2, 'i'), (3, 'p')]`. Defina a função `nat_zip` usando a função `zip`.

6. (20%) Implemente uma função `quadrados` que dada uma lista de inteiros `xs` calcule uma outra lista que contém os quadrados dos elementos de `xs`. Por exemplo `quadrados [2,10,1] = [4,100,1]`.

(a) Defina a função `quadrados` recursivamente.

(b) Defina a função `quadrados` usando listas em compreensão.

7. (10%) Defina uma função `partes`, que dado um inteiro positivo `n`, calcula todas as representações de `n` como somas de inteiros positivos. Por exemplo `partes 4 = [[1, 1, 1, 1], [1, 1, 2], [1, 3], [2, 2], [4]]`. Note que o resultado inclui `[1, 3]` mas não `[3, 1]`. (Sugestão: comece por definir uma função `crescente`, que verifica se uma lista está em ordem crescente).