Aula Teórico-prática 5

Programação Funcional

LEI 1º ano

1. (a) Defina as funções `div` e `mod` que calculam respectivamente a divisão e o resto da divisão inteira de um número por outro.

   (b) Defina uma função que calcula simultaneamente estes dois resultados: `divMod :: Int -> Int -> (Int,Int)`.

   Note que apesar de poder ser definida à custa das outras duas, i.e., usando a definição

   ```
   divMod x y = (div x y, mod x y)
   ```

   nessa definição há trabalho redundante que pode ser evitado. Apresente uma definição alternativa onde não haja duplicação de trabalho.

2. A função `splitAt :: Int -> [a] -> ([a],[a])`, já predefinida no Prelude, poderia ser definida pela seguinte equação:

   ```
   splitAt n l = (take n l, drop n l)
   ```

   no entanto nessa definição há uma duplicação de trabalho, dado que se fazem duas travessias da lista. Apresente uma versão alternativa para esta função que faça apenas uma travessia da lista.

3. O algoritmo *merge sort*, de ordenação de listas, pode ser descrito do seguinte modo:

   1. *Parte-se a lista em duas listas de tamanho igual (ou quase).*
   2. *Ordenam-se as duas sublistas (geradas em 1.)*
   3. *Fundem-se as duas listas já ordenadas de forma a que a lista resultante fique ordenada.*

   (a) Defina uma função que parte uma lista em duas listas de tamanho igual (ou quase).

   (b) Defina uma função que, dadas duas listas ordenadas, funde as listas numa lista ordenada.

   (c) Defina a função `msort :: [Int] -> [Int]` que ordena uma lista de inteiros, segundo o algoritmo merge sort.

4. Averigue o comportamento das funções `words :: String -> [String]` e `unwords :: [String] -> String`, e defina-as.