Aula Teórico-prática 5

Programação Funcional

LEI 1º ano (2006/2007)

1. Considere a seguinte definição da função `unzip`.

   ```
   unzip :: [(a,b)] -> ([a],[b])
   unzip l = (map fst l, map snd l)
   ```

   Apresente uma definição alternativa que não percorra duas vezes a lista argumento.

2. (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)
   ```

   essa definição não é muito *eficiente*.

3. (a) Apresente definições das funções sobre listas `take, drop :: Int -> [a] -> [a]`.

   (b) Defina uma função que calcula simultaneamente estes dois resultados: `splitAt:: Int -> [a] -> ([a],[a])`. Note que apesar de poder ser definida à custa das outras duas, i.e. usando a definição

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

   essa definição não é muito *eficiente*.

4. (a) Apresente definições das funções sobre listas `takeWhile, dropWhile :: (a->Bool) -> [a] -> [a]`.

   (b) Defina uma função que calcula simultaneamente estes dois resultados: `break:: (a-> Bool) -> [a] -> ([a],[a])`. Note que apesar de poder ser definida à custa das outras duas, i.e. usando a definição

   ```
   break p l = (takeWhile p l, dropWhile p l))
   ```

   essa definição não é muito *eficiente*.

5. Redefina as funções `lines`, `words` e `paragrafos` da aula anterior de forma a usar a função `break`.