Aula Teórico-prática e Prática 4

Programação Funcional

LCC 1º ano

1. Defina funções `primeiros::[(a,b)] -> [a]` e `segundos::[(a,b)] -> [b]` que extraem as primeiras e segundas componentes de uma lista de pares.

   Considere agora a seguinte definição da função `unzip`.

   ```
   unzip :: [(a,b)] -> ([a],[b])
   unzip l = (primeiros l, segundos 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. Defina uma função que `group :: [a] -> [[a]]` que agrupa os elementos consecutivamente iguais de uma lista. Por exemplo,

   ```
   agrupa [1,4,4,3,2,2,2,1,1,1]=[[1],[4,4],[3],[2,2,2],[1,1,1]]
   ```

4. Usando a função da alínea anterior, defina uma função `comprime :: [a] -> [(a,Int)]` de tal forma que

   ```
   comprime [1,4,4,3,2,2,2,1,1,1]=[(1,1),(4,2),(3,1),(2,3),(1,3)]
   ```

5. Defina a função `descomprime :: [(a,Int)] -> [a]` inversa da anterior.