Paradigmas de Programação

Fabrício Olivetti de França

03 de Julho de 2018

Exercícios de recursão

1. Defina as seguintes funções recursivas:

```haskell
-- Decide se todos os valores lógicos de uma lista são True
and :: [Bool] -> Bool

-- Concatena uma lista de listas
concat :: [[a]] -> [a]

-- Produz uma lista com n valores idênticos
replicate :: Int -> a -> [a]

-- Seleciona o enésimo elemento de uma lista
(!!) :: [a] -> Int -> a

-- Verifica se um valor é um elemento de uma lista
elem :: Eq a => a -> [a] -> Bool
```

2. Defina uma função recursiva denominada `merge :: Ord a => [a] -> [a] -> [a]` que junta duas listas ordenadas, resultando em uma única lista ordenada:

```
> merge [2,5,6] [1,3,4]
[1,2,3,4,5,6]
```

Não use funções já criadas para ordenação.

3. Usando a função anterior, defina a função `msort :: Ord a => [a] -> [a]` que implementao algoritmo Merge Sort seguindo as regras:

- Uma lista vazia ou um *singleton* já está ordenado
- Qualquer outra lista é ordenada dividindo a lista em duas metades, ordenando-as com `msort` e juntando com a função `merge`
- Nota: crie uma função `metade :: [a] -> ([a],[a])` que divide uma lista ao meio.

4. Verifique as propriedades do algoritmo de ordenação com o QuickCheck