Época Especial de Programação Funcional – 1º Ano, LEI / LCC / LEF
18 de Julho de 2023 (Duração: 2 horas)

1. Apresente uma definição recursiva das seguintes funções (pré-definidas) sobre listas:

   (a) Apresente uma definição recursiva da função `tails :: [a] -> [[a]]` que calcula a lista dos sufixos de uma lista.
   Por exemplo, `tails [1,2,3]` corresponde a `[[1,2,3],[2,3],[3],[]]`.

   (b) `isPrefixOf :: Eq a => [a] -> [a] -> Bool` que testa se uma lista é prefixo de outra. Por exemplo, `isPrefixOf [10,20] [10,20,30]` corresponde a `True` enquanto que `isPrefixOf [10,30] [10,20,30]` corresponde a `False`. [] [10] [10,20] [10,20,30]

2. Considere a sequinte definição para representar matrizes: `type Mat a = [[a]]`

   Por exemplo, a matriz $\begin{bmatrix} 1 & 2 & 3 \\ 0 & 4 & 5 \\ 0 & 0 & 6 \end{bmatrix}$ seria representada por `[[1,2,3], [0,4,5], [0,0,6]]`

   (a) Defina uma função `zeros :: Num a => Mat a -> Int` que conta o número de zeros presentes na matriz.

   (b) Defina uma função `addMat :: Num a => Mat a -> Mat a -> Mat a` que adiciona duas matrizes. De preferência, faça-o com a ajuda de uma função de ordem superior.

3. Considere a seguinte declaração de tipos que associa a cada filme uma lista de avaliações com classificações do filme. Para isso, definiram-se os tipos de dados `Avaliacao` e `FilmesAval`.

```
data Avaliacao = NaoVi
               | Pontos Int        -- pontuação entre 1 e 5
  deriving (Eq)
type Filme = String
type FilmesAval = [(Filme,[Avaliacao])]
```

   (a) Declare `Avaliacao` como instância da classe `Ord`. Considere que `NaoVi` é a menor avaliação de todas.

   (b) Defina a função `avalia :: FilmesAval -> IO FilmesAval` que lê o título de um filme e a sua avaliação e, caso o filme exista, acrescenta essa avaliação à base de dados de filmes. A escala de classificação deve ser de 1 a 5.

4. Considere o seguinte tipo para representar árvores irregulares (*rose trees*).

```
data RTree a = R a [RTree a]
```

   Defina a função `percorre :: [Int] -> RTree a -> Maybe [a]` que recebe um caminho e uma árvore e dá a lista de valores por onde esse caminho passa. Se o caminho não for válido a função deve retornar `Nothing`. O caminho é representado por uma lista de inteiros (1 indica seguir pela primeira sub-árvore, 2 pela segunda, etc)