IFCE - Campus Maracanaú **Ciência da Computação**
Linguagens de Programação **Prof. Thiago Alves**

7ª Lista de Exercícios

Aluno(a): _____ Matrícula: _____

1. Defina uma função recursiva `flat :: [[a]] -> [a]` que recebe uma lista de listas e retorna a lista de todos os elementos das sublistas. Por exemplo:

   ```
   > flat [[1,2],[4,2,1]]
   [1,2,4,2,1]
   ```

2. Defina uma função recursiva `merge :: Ord a => [ a ] -> [ a ] -> [ a ]` que junta duas listas ordenadas e retorna uma única lista ordenada. Por exemplo:

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

3. Usando `merge`, defina uma função recursiva `msort :: Ord a => [ a ] -> [ a ]` que implementa o merge sort, no qual a lista vazia e a lista unitária já estão ordenadas, e qualquer outra lista é ordenada da seguinte forma: separar a lista em duas metades, ordena as duas metades e depois usa o `merge` para juntar as duas metades.

4. Mostre como a compreensão de listas `[f x | x <- xs, p x]` pode ser definida usando as funções de alta-ordem `map` e `filter`.

5. Defina a função `flat :: [[a]] -> [a]` usando `foldr`.

6. Uma empresa tem várias filiais e quer saber o gasto total com o salário dos funcionários. Defina uma função `gastoEmpresa :: [[Float]] -> Float` em que cada sublista representar a lista de salários de uma filial e retorna o gasto total com os salário. Em seguida, a empresa quer dar uma aumento de 10% para os funcionários que ganham menos de R$1000,00 e um aumento de 5% para o restante. Faça uma função `aumentarSal :: [[Float]] -> [[Float]]` para realizar essa operação. Para finalizar, a empresa gostaria de saber qual vai ser o novo gasto com os salários depois do aumento. Para isso, defina a função `novoGasto :: [[Float]] -> Float` usando composição de funções.