# 1º Teste

Programação Funcional – 1° Ano, LEI / LCC / MIEF
27 de Novembro de 2013

Duração: 90 min

Considere as seguintes declarações de tipos para representar uma colecção de albuns de música.

```
type Album = (Titulo,Artista,Ano,[Musica])
type Musica = (Nome,Int)        -- (nome da música, duração em segundos)
type Titulo = String
type Nome = String
type Artista = String
type Ano = Int
```

1. Defina a função `doArtista :: [Album] -> Artista -> [(Titulo,Ano)]` que, dado o nome de um artista, calcula a lista dos seus albuns.

2. Defina a função `conta :: [Artista] -> [Album] -> [(Artista,Int)]` que, dada uma lista de artistas e uma colecção de albuns, calcula quantos albuns de cada artista da lista existem na colecção.

3. Apresente uma definição alternativa da seguinte função usando recursividade explícita em vez de funções de ordem superior.

   ```
   fun :: Artista -> [Album] -> [(Titulo,Int)]
   fun x l = map aux (filter (\(_,a,_,_)->x==a) l)
     where aux (t,_,_,m) = (t, sum (map snd m))
   ```

4. Defina a função `maisAntigos :: [Album] -> [Album]` que, dada uma colecção de albuns, devolve lista dos albuns mais antigos da colecção, i.e., a lista dos albuns do ano mais antigo presente na colecção.

5. Defina uma função `covers :: [Album] -> [(Nome,[(Artista,Ano)])]` que, para todas as músicas de uma colecção que são interpretadas por **mais do que um** artista, determina a lista dos intérpretes (e ano correspondente) dessa música.