



Disciplina: Programação 3 - Programação Funcional

Professor: Emanuel Barreiros

Assunto: Compreensão de Listas

Resumo: Utilize a linguagem de programação Haskell para resolver os problemas desta lista. Esta lista aborda o conceito de compreensão de listas. O conteúdo teórico relacionado a esta lista de exercícios pode ser encontrado aqui:

<https://emanoelbarreiros.github.io/funcional/haskell-5>

- 1) Usando compreensão de listas, forneça uma expressão que calcula a soma $1^2 + 2^2 + \dots + 100^2$ dos quadrados dos primeiros 100 números inteiros.
- 2) Suponha que um plano de coordenadas de tamanho $m \times n$ é dado pela lista de todos os pares (x,y) de inteiros tal que $0 \leq x \leq m$ e $0 \leq y \leq n$. Usando compreensão de listas, defina a função `grid :: Int -> Int -> [(Int,Int)]` que retorna o plano de coordenadas de um dado tamanho. Por exemplo:

```
> grid 1 2  
[(0,0), (0,1), (0,2), (1,0), (1,1), (1,2)]
```
- 3) Usando compreensão de listas e a função `grid` definida na questão anterior, defina uma função `quadrado :: Int -> [(Int,Int)]` que retorna um plano de coordenadas quadrado de tamanho n , excluindo a diagonal principal $(0,0)$ a (n,n) . Por exemplo:

```
> quadrado 2  
[(0,1), (0,2), (1,0), (1,2), (2,0), (2,1)]
```
- 4) De maneira similar à função `length`, mostre como a função `replicate :: Int -> a -> [a]` que produz uma lista de elementos idênticos pode ser definida usando compreensão de listas. Exemplo:

```
> replicate 3 True  
[True, True, True]
```
- 5) Uma tupla (x,y,z) de inteiros positivos é Pitagoreana se satisfaz a equação $x^2 + y^2 = z^2$. Usando compreensão de listas com três geradores, defina a função `pitag :: Int -> [(Int, Int, Int)]` que retorna uma lista de todas as tuplas que satisfazem a condição estabelecida e cujos componentes são menores ou iguais a um dado limite. Exemplo:

```
> pitag 10  
[(3,4,5), (4,3,5), (6,8,10), (8,6,10)]
```
- 6) Um inteiro positivo é perfeito se ele é igual à soma de todos os seus fatores, excluindo o próprio número. Usando compreensão de listas e a função `fatores`,



defina a função `perfeitos :: Int -> Int` que retorna a lista de todos os números perfeitos menores que um limite informado como argumento. Exemplo:

```
> perfeitos 500  
[6, 28, 496]
```

- 7) Mostre que a compreensão de lista `[(x,y) | x <- [1,2], y <- [3,4]]`, com dois geradores, pode ser representada usando duas compreensões de lista, cada uma com apenas um gerador. Dica: Procure usar a função [`concat`](#).
- 8) Redefina a função `posicoes` usando a função `buscar`, disponível em <https://emanoelbarreiros.github.io/funcional/haskell-5#a-fun%C3%A7%C3%A3o-zip>
- 9) O produto escalar de duas listas de inteiros `xs` e `ys` de tamanho `n` é dado pelo produto dos inteiros em posições correspondentes:

$$\sum_{i=0}^{n-1} (xs_i * ys_i)$$

Dica: Procure usar a função `zip`.