Lista de Exercícios de Programação Funcional em Haskell

Instruções: Resolva todos os exercícios sem utilizar `let-in` ou `where`. Apenas definições diretas de funções são permitidas.

# Exercícios Fáceis

1. Defina uma função `somaQuadrados` que receba dois números e retorne a soma dos quadrados desses números.
2. Defina uma função `ehPar` que receba um número inteiro e retorne `True` se ele for par e `False` caso contrário.
3. Defina uma função `maiorDeDois` que receba dois números e retorne o maior deles.
4. Defina uma função `areaCirculo` que receba o raio de um círculo e retorne a sua área (use `pi`).
5. Defina uma função `segundoElemento` que receba uma lista e retorne o segundo elemento dela.

# Exercícios Médios

1. Defina uma função `produtoLista` que receba uma lista de números e retorne o produto de todos os seus elementos.
2. Defina uma função `todosPositivos` que receba uma lista de inteiros e verifique se todos os elementos são positivos.
3. Defina uma função `contaVogais` que receba uma string e conte quantas vogais ela possui.
4. Defina uma função `replicaElemento` que receba um número n e um elemento, e crie uma lista com o elemento repetido n vezes.
5. Defina uma função `potenciaDeDois` que receba um número inteiro n e retorne 2 elevado a n.

# Exercícios Desafiadores

1. Defina uma função `fatorialDuplo` que calcule o fatorial de um número considerando apenas números ímpares (ex: 7!! = 7\*5\*3\*1).
2. Defina uma função `intercalaListas` que receba duas listas e intercale seus elementos (caso uma lista seja maior, ignore os elementos restantes).
3. Defina uma função `somaPrefixos` que, dada uma lista de números, calcule a soma de todos os seus prefixos. Exemplo: somaPrefixos [1,2,3] = (1) + (1+2) + (1+2+3) = 10.
4. Defina uma função `numeroPerfeito` que receba um número e verifique se ele é perfeito (soma dos divisores próprios igual ao próprio número).
5. Defina uma função `palindromo` que verifique se uma string é um palíndromo (ex: "arara", "osso").