1. Usando a função filter declare uma lista infinita com os números naturais que são simultaneamente múltiplos de dois e três.

2. Declare uma função que recebe uma lista de Strings e retorna uma lista de duplas onde, o primeiro elemento é a String recebida, e e o segundo é o tamanho desta String.

   Ex: contaPalavras ["Programacao", "Funcional"] => [("Programacao",11),("Funcional",9)]

3. Declare uma função que recebe um número n e retorna pares de valores distintos (x, y), tal que: 1<=x, y<=n. Utilize filter na construção.

   Ex: pares 3 => [(1, 2), (1, 3), (2, 1), (2, 3), (3, 1), (3, 2)]

4. Declare uma função que recebe uma lista de caracteres, e retorne a mesma lista porém removendo as letras maiúsculas.

   Ex: semMaiusc "Programacao Funcional" => "rogramacao uncional"

5. Declare uma função que recebe uma lista de listas, e remove toda a ocorrência de lista vazia desta lista de listas.

   Ex: semListasVazias [[1, 2], [], [1, 3]] => [[1,2], [1, 3]]

6. Declare uma função que recebe uma lista com pares de inteiros e retorna outra lista que contém a soma de cada par.

   Ex: somaPares [(2,3),(1,8)] => [5,9]

7. Declare uma função que recebe uma lista de números inteiros e retorna uma dupla de listas ([a],[b]), onde [a] contempla os elementos ímpares, e [b] os elementos pares.

   Ex: separaParImpar [1,2,3,4,5,6,7] => ([1,3,5,7],[2,4,6])

8. Declare uma função com comportamento equivalente a função take da biblioteca prelude de Haskell, ou seja, deve receber um número inteiro n e uma lista, e retornar os n primeiros desta lista. Utilize as funções de ordem superior.

   Ex: take' 2 [1,2,3,4] => [1,2]

9. Declare uma função, usando foldr, que receba um lista de valores booleanos e retorne a True se todos elementos da lista forem True, a função deve retornar False caso contrário.

10. Declare uma função que recebe um número inteiro n, e retorna a soma do quadrado dos n primeiros números. Defina duas versões desta função, uma utilizando map e outra utilizando fold.

    Ex: somaQuadrado 4 = 1^2 + 2^2 + 3^2 + 4^2 => 30

11. Declare a sua versão da função length disponível na biblioteca de Haskell Prelude, porém utilizando a função de ordem superior foldr.

    Ex: length' "Programacao Funcional" => 21

12. Declare uma função que recebe uma lista de inteiros, e retorna o menor número inteiro desta lista. Defina 2 versões desta implementação, uma utilizando foldr e outra utilizando foldl.

    Ex: minlista [2,5,1,8] => 1

13. Declare uma função que recebe uma lista de Strings e retorna uma lista de booleanos tal que, True é o enésimo elemento da lista de inteiros se o enésimo elemento da lista de Strings tem um número de caracteres par, e False caso seja ímpar.

    Ex: paridade ["um","dois","tres","quatro","cinco"] => [True,True,True,True,False]

14. Declare uma lista infinita de números primos, use o crivo de Eratóstenes.