Ordem superior

1. Nos exercícios a seguir, **escreva primeiramente uma função recursiva**. Em seguida, reescreva essa função utilizando função de ordem superior. **Você pode usar as funções nativas map, filter e foldr**.

   (a) Função **pares :: [Int] -> [Int]** que remove todos os elementos ímpares de uma lista

   (b) Função **rm_char :: Char -> String -> String** que remove todas as ocorrências de um caractere em uma string.

   (c) Função **acima :: Int -> [Int] -> [Int]** que remove todos os números menores ou iguais a um determinado valor.

   (d) Função **produto :: Num a => [a] -> a** que computa o produto dos números de uma lista.

   (e) Função **concatena :: [String] -> String** que junta uma lista de strings em uma única string.

2. Faça a função **ssp** que considera uma lista de inteiros e devolve a soma dos quadrados dos elementos positivos da lista.

3. Defina a função **sumsq** que considera um inteiro n como argumento e devolve a soma dos quadrados dos n primeiros inteiros. Ou seja:

   ```
   *Main > sumsq 4
   30
   ```

   pois $1^2 + 2^2 + 3^2 + 4^2 = 30$.

4. Uma função que devolva o valor da soma dos comprimentos de cada string (elemento) da lista. Isto é, a soma total dos comprimentos da lista de entrada.

5. Faça uma função que separe caracteres de números em uma string de entrada. O retorno é uma tupla, em que no primeiro argumento esteja a sequência de caracteres (string), e no segundo argumento uma sequência de inteiros. **Dica:** Utilize isAlpha e isDigit, presentes em Data.Char.
   Por exemplo:

   ```
   > separa "aA29bB71"
   ("aAbB ", "2971")
   ```