IF686 - Paradigmas de Linguagens Computacionais
Programação Funcional (Haskell)
Segunda Lista de Exercíos

1. (0,2 points) Use as funções sqrt :: Float -> Float, que retorna a raiz quadrada de um valor positivo de tipo Float, e as funções filter , map e foldr para definir uma função composta que retorna a soma das raízes quadradas dos valores positivos de uma lista de valores de tipo Float. Os valores negativos (e os iguais a zero) devem ser desconsiderados.

   Exemplos

   ```
   somaSqrt [(-5), 7, 5, (-9) ] ==> 4.8818192885643805
   somaSqrt [ 7, 5 ] ==> 4.8818192885643805
   somaSqrt [(-5), 7, 5, (-9), 0 ] ==> 4.8818192885643805
   ```

2. (0,2 points) Considere unzip:: [(a,b)] -> ([a],[b]), definida como:

   ```
   unzip :: [(a,b)] -> ([a],[b])
   unzip [] = ([],[])
   unzip ((a, b): xs) = (a:as, b:bs)
    where
      (as, bs) = unzip xs
   ```

   Exemplo

   ```
   unzip [(1,2), (3,4), (5,6)] ==> ([1,3,5],[2,4,6])
   ```

   Defina uma nova versão da função unzip, usando foldr.

3. A partir dos tipos

   ```
   type Texto = String
   type Id = String
   type DataHoraPub = Int
   ```

   podemos descrever *posts* e *thread* em uma rede social com os tipos

   ```
   data Post = Post (Id, DataHoraPub) Texto deriving (Show, Eq)
   data Thread = Nil | T Post (Thread)
   ```

   (a) (0,2 points) Estabeleça que Thread é instância da classe Show de modo que a *thread* (T (Post ("Joao", 1) "asdf") (T (Post ("Marco", 2) "qwer") Nil)) é exibida da seguinte forma: "(Joao 1 asdf)(Marco 2 qwer)"

   Implemente a função show para que a exibição se dê como solicitado.

   (b) (0,2 points) Defina a função inserirPost que, dado um *post* e uma *thread*, devolve uma nova *thread* com o novo *post*.

   Exemplo

inserirPost (Post ("Marco", 2) "qwer") (T (Post ("Joao", 1) "asdf") Nil)

=

(Marco 2 qwer)(Joao 1 asdf)

(c) (0,2 points) Defina a função threadToList :: Thread -> [Post] que transforma uma *thread* em uma lista de *posts*.

Exemplo

threadToList (T (Post ("Joao", 1) "asdf") (T (Post ("Marco", 2) "qwer") Nil))

=

[Post ("Joao",1) "asdf",Post ("Marco",2) "qwer"]

(d) (0,2 points) Defina a função listToThread :: [Post] -> Thread que transforma uma lista de *posts* em um *thread*.

Exemplo

listToThread [Post ("Joao",1) "asdf",Post ("Marco",2) "qwer"]

=

(Joao 1 asdf)(Marco 2 qwer)

(e) (0,2 points) Defina a função removerPost :: (Id, DataHoraPub) -> Thread -> Thread que remove um *post* identificado pelo par (Id, DataHoraPub) de uma *thread*. Utilize a função filter na implementação.

Exemplo

removerPost ("Marco", 2) (T (Post ("Joao", 1) "asdf") (T (Post ("Marco", 2) "qwer") Nil))

=

(Joao 1 asdf)