Solution Types and Polymorphism

Instructions:
Solutions of the exercises are to be delivered before Thursday, the 22th of March at 10:15AM.
Solutions should be placed in a separate folder with the name “Assignment04”.
Please submit answers to all the exercises in one text file.

Exercise 1 (3 points)
Infer types of the functions factors, isPerfect and insert and say whether they are monomorphic or polymorphic functions. Justify your answer.

• mod :: Int -> Int -> Int
  factors n = [x | x <- [1..n-1], mod n x == 0 ]
  isPerfect n = sum (factors n) == n

• insert _ n [] = [n]
  insert 0 n l = n:l
  insert i n (x:xs) = x : insert (i-1) n xs

Answer:
factors :: Int -> [Int]
since both n and x are arguments of the function mod which accepts only the Int arguments

isPerfect :: Int -> Bool
since n is an argument of the function factors which accepts only the Int arguments,
and == :: Eq a => a -> a -> Bool

Both functions are monomorphic.

insert :: Int -> [a] -> [a]
since
insert _ n l = [n] => insert :: a->b->c->[b]
insert 0 n l = n:l => insert :: Int->b->[b]->[b]
The insert function is polymorphic.

Exercise 2 (3 points)
Infer the type of the following function and explain each of the steps.
f1 f x
  | f x < 0 = []
  | otherwise = x : (f1 f (f x))
**Answer:**

\[
f_1 :: \quad a \to b \to c \quad \text{since } f_1 \text{ takes two arguments and returns something}
\]

\[
a \to b \to [d] \quad \text{since } c \text{ is of type list}
\]

\[
(e \to g) \to b \to [d] \quad \text{since } f \text{ takes one argument}
\]

\[
(\text{Ord } h \Rightarrow e \to h) \to b \to [d] \quad \text{since } f \text{ takes } x \text{ as an argument}
\]

\[
(\text{Ord } h \Rightarrow b \to h) \to b \to [d] \quad \text{since } f \text{ takes } f \ x \text{ as an argument}
\]

\[
(\text{Ord } b \Rightarrow b \to b) \to b \to [b] \quad \text{since the result of } f_1 \text{ is the list whose head is } x
\]

The result is:

\[\text{:t } f_1\]

\[f :: (\text{Ord } a \Rightarrow a \to a) \to a \to [a]\]

**Optional Haskell exercise (2 points)**

Write a function `deleteRepetitions l` which deletes all consecutive repetitions of elements in the list `l`. For example, `deleteRepetitions [4, 5, 5, 2, 11, 11, 11, 2, 2]` would return as the result `[4, 5, 2, 11, 2]`. **No built-in function for working with lists may be used. Only pattern matching is allowed.**

**Answer:**

\[
deleteRepetitions [] = []
deleteRepetitions (head:\[]) = [head]
deleteRepetitions (first:second:tail) =
  if first == second
  then deleteRepetitions (second:tail)
  else first : deleteRepetitions (second:tail)
\]