Compitino 2 di Linguaggi 2015-16 23 ottobre 2015

1) Inferire il tipo della seguente funzione:

f x [] []= []

f x (a:b) (c:d)= (x a c):(f x b d)

f x (a:b) [] = (x a a):(f x b [])

f x [] (c:d) = (x c c):(f x [] d)

Gestire il fatto che si usa il pattern matching con una piccola prova separata dall'inferenza di f. Numerare i nodi degli alberi costruiti per l'inferenza e per ogni vincolo mostrare da quale nodo esso proviene.

2) Applicare l'algoritmo di unificazione sull'insieme di vincoli prodotto da uno dei casi della funzione f del punto precedente. Non il primo caso. Basta fare solo qualche passo. Cercate di mostrare l'applicazione dei passi 1 e 5 dell'algoritmo.

3) Spiegare (brevemente) la dimostrazione di correttezza dell'algoritmo di unificazione.

4) Questa domanda riguarda i programmi Haskell:

a) dato il tipo: data Tree a = EmptyTree | Node a (Tree a) (Tree a) deriving (Eq,Show);

definire una funzione treeinsert (con tipo Ord a=>a->Tree a -> Tree a) capace di costruire un albero Binario di Ricerca di tipo Tree a.

b) Usare foldr per costruire un albero binario di ricerca che abbia nei nodi interni delle liste [Int] e che usa una lista di liste [[Int]].

c) definire Tree come un'istanza di Functor e quindi definire la funzione fmap caratteristica di Functor in modo da poter "spalmare" sull'albero prodotto in (b) funzioni adatte a venire applicate su liste di Int, come (1:) oppure (map (1+))