p1q1
LCP

Foundations of Computer Science (10 Mark Question) 2000

Code the curried function `exf`, which takes as arguments the function f and the list l. The result must consist of those elements x of l such that $f(x)$ is also a member of l. The elements of the result must be distinct from each other but may appear in any order. For example, if $f(x) = x + 1$ and $l = [9, 3, 2, 2, 8]$ then the result should be $[2, 8]$ or $[8, 2]$. [9 marks]

State, with justification, the type of `exf`. [1 marks]

Model Answer

Here is a solution. ~~It is not essential to use foldr, but it avoids the use of very expensive appending of the results for the various subtrees. Note that even if a node is accepted, its subtrees are still examined.~~

```
(*delete all occurrences of y in the list*)
fun delete y [] = []
  | delete y (x::xs) = if x=y then delete y xs
                       else   x :: delete y xs;

fun exists p [] = false
  | exists p (x::xs) = p x orelse exists p xs;

fun exf f l =
  let fun aux [] = []
        | aux (x::xs) =
            if exists (fn y => y = f x) l
            then x :: delete x (aux xs)
            else aux xs
  in  aux l  end;
```

The type is `(''a -> ''a) -> ''a list -> ''a list`. The type variable `''a` has two quote marks because equality testing is performed. The nesting of the arrows is typical of a curried function. The argument and result lists have the same type.

Also a neat solution by

```
fun setof [] = []
  | setof (x::xs) = if member (x, xs) then
                      setof xs
                    else x :: setof xs;

fun inter1 ([], ys) = []
  | inter (x::xs, ys) = if member (x, ys)
                        then x :: inter (xs, ys)
                        else inter (xs, ys);

fun exf f l = setof (inter l (map f l));
```