Name: **Matr.nr.:**

This exam consists of five exercises. Please motivate your answers unless mentioned otherwise. The number of points per question is indicated in the left margin. The exam is passed if you get 25 out of 50 points or more.

[5] 1. Given the function

```
let rec product l = match l with
 | [] -> 1
 | x::xs -> x * (product xs)
;;
```

Give an optimized definition of this function that uses

(a) Tail recursion

(b) The fact that $n * 0 = 0$ for any n

[5] 2. Consider the following functions:

```
let rec map1 f l = match l with
 | [] -> []
 | x::xs -> (f x)::(map1 f xs)
;;
let rec map2 f l = match l with
 | [] -> []
 | x::xs -> let y = f x in y::(map2 f xs)
;;
let rec maprev f a l = match l with
 | [] -> a
 | x::xs -> maprev f ((f x)::a) xs
;;
let rec rev a l = match l with
 | [] -> a
 | x::xs -> rev (x::a) xs
;;
let map3 f l = rev [] (maprev f [] l);;
let map4 f l = maprev f [] (rev [] l);;
```

Please fill the following table with A,P,D meaning:

A Always equal, regardless of the choice for f and l.

P equal for Pure functions only: if f has side effect they can be different

D Different: you can find a list and a pure (side-effect free) function that evaluate to different results.

	map1	map2	map3	map4
map1	A			
map2	X	A		
map3	X	X	A	
map4	X	X	X	A

[15] 3. Consider the functions

```
let rec uniq1 l = match l with
 | [] -> []
 | x1::x2::xs when x1=x2 -> uniq1 (x2::xs)
 | x::xs -> x::(uniq1 xs)
;;
let rec drop x l = match l with
 | y::ys when x = y -> drop x ys
 | _ -> l
;;
let rec uniq2 l = match l with
 | [] -> []
 | x::xs -> x::(uniq1 (drop x xs))
;;
```

Prove that for all lists l, we have that

$$\text{uniq1}\, l = \text{uniq2}\, l$$

Hint: distinguish between lists with one element and lists with more than one element.

[10] 4. Use the proof system for types to show that

$$\vdash \lambda f\, x.f(f\, x) : (\alpha \to \alpha) \to (\alpha \to \alpha)$$

[15] 5. Infer the type of

```
let rec filter p l = match l with
 | [] -> []
 | x::xs when p x -> x :: (filter p xs)
 | x::xs -> (filter p xs)
;;
```