Exercises Week 10

1. [1+1 POINTS] Consider the type `'a tree` and the mirroring function `mirror` (see the slides of `w10-1x2.pdf` or `w10-1x2.pdf`).

   (a) `size` t returns the number of nodes in the tree t. Implement `size`.

   (b) Show that `size` (`mirror` t) $=$ `size` t for all trees t.

2. [2 POINTS] Consider the functions

   ```
   let rec rev = function
     | [] -> []
     | x :: xs -> rev xs @ [x]

   let rec rev_append l1 l2 =
     match l1 with
     | [] -> l2
     | a :: l -> rev_append l (a :: l2)

   let rev' l = rev_append l []
   ```

   Show that `rev` $l =$ `rev`$'$ l for all lists l.

3. [1+1 POINTS]

   (a) Define a tail recursive version `length'` of `length`:

   ```
   let rec length = function
     | [] -> 0
     | x :: xs -> 1 + length xs
   ```

   (b) Prove that `length` $l =$ `length'` l holds for all lists l.

4. [1+1 POINTS] Consider the following type for expressions:

   ```
   type e = Var of string | T | F
          | Not of e | And of e * e | Or of e * e
   ```

   (a) Use the following 10-rule rewrite system to implement `simplify`.

$\texttt{Not T} \to \texttt{F}$	$\texttt{And}\ (\texttt{T}, e) \to e$	$\texttt{Or}\ (\texttt{T}, e) \to \texttt{T}$
$\texttt{Not F} \to \texttt{T}$	$\texttt{And}\ (e, \texttt{T}) \to e$	$\texttt{Or}\ (e, \texttt{T}) \to \texttt{T}$
	$\texttt{And}\ (\texttt{F}, e) \to \texttt{F}$	$\texttt{Or}\ (\texttt{F}, e) \to e$
	$\texttt{And}\ (e, \texttt{F}) \to \texttt{F}$	$\texttt{Or}\ (e, \texttt{F}) \to e$

   (b) `substitute` x e_1 e_2 substitutes the expression e_1 into the variable x in the expression e_2. Implement it.

   ```
   # simplify (And (Or (Not T, Not (Var "x")),
                     And (Var "y", (Or (T, F)))));;
   - : e = And (Not (Var "x"), Var "y")

   # substitute "y" T (And (Not (Var "x"), Var "y"));;
   - : e = And (Not (Var "x"), T)
   ```

Submit `yourMatrNr.ml` before 23:59 on **January 11**.

```
(* 1(a) *)
let rec size = function ..

(* 1(b) mirror (size t) = size t for all trees t.
Proof by induction on t.
Base case:
...
*)
```