Homework 2: Type Reconstruction and Typed Lambda Calculus

Consider this homework to be "due" on September 24.

1. How would you extend ML's type reconstruction algorithm and the type inference rules discussed in Cardelli's paper to deal with record types? (Use the syntax of SML for records and their types.)

2. In lecture it was mentioned that ML's type checker cannot handle recursive type definitions of the following form.

```
type 'a stream = unit -> ('a * ('a stream))
```

   Explain why this is a problem.

3. Some exercises with the formal system of the typed lambda calculus. Prove the following.

   (a) If $H, x : s \vdash e : t$ and $H \vdash e' : s$, then $H \vdash [e'/x]e : t$.

   (b) If $H \vdash e : s$ and x is not in the domain of H, then $H, x : t| - e : s$.
   The above lemma allows one to add type assumptions. State a symmetric property that allows one to delete type assumptions and prove it.

   (c) State and prove a (derived) equational rule that shows that equality is preserved under consistent renaming of free variables.

   (d) If x is not in $\mathbf{Fv}(e)$ and $H \vdash e : t$, then $[\![H, x : s \rhd e : t]\!]\rho[d/x] = [\![\rhd e : t]\!]\rho$.

   (e) Derive the type of $\lambda f : \mathbf{o} \to \mathbf{o} \,.\, \lambda x : \mathbf{o} \,.\, f(x)$ in the empty environment.
   Let t be the type you derived above. Show that

$$\not\vdash \lambda f : \mathbf{o} \to \mathbf{o} \,.\, \lambda x : \mathbf{o} \,.\, f(f(x)) = \lambda f : \mathbf{o} \to \mathbf{o} \,.\, \lambda x : \mathbf{o} \,.\, f(x) : t.$$