Functional Programming
Exercises Week 12
(for January 20, 2017)

1. Read Chapter 10 of the lecture notes.

2. Consider the following type

```
type exp = Add of (exp * exp) | Num of int | Var of int
```

representing expressions with the following grammar,

$$e := e + e \mid n \mid x_i$$

where n is a constant number, and x_i is a variable (represented by the index $i \geq 0$). Write a function

```
eval : int Lazy.t list Lazy.t -> exp -> int Lazy.t
```

that evaluates an expression of type `exp`, where the value of x_i is given by the i-th element of the given list.

3. ($\star$) Continuing Exercise 2, write a function

```
resolve : exp list -> int list
```

that takes a list of expressions $e_0, \ldots, e_n$ using only variables x_0 to x_n and unfolds the equations

$$\begin{aligned} x_0 &:= e_0 \\ &\ldots \\ x_n &:= e_n \end{aligned}$$

You may assume that these equations do not have a cycle.

Hint: This can be done using only `List.map`, `eval`, `Lazy.force` and builtin syntax (this includes the **`lazy`** keyword).

Hint 2: How can one obtain a value of type `int Lazy.t list Lazy.t`?

4. Solve the type inference problem $P \triangleright e : \alpha_0$ for the core ML expression

$$e \equiv \lambda f x.f\ (f\ 0\ x)\ \mathsf{true}$$

5. Exercise 9.7