Today

• Mutually recursive functions
• Modules
• Higher-order functions
Mutual Recursion

What if we need function \( f \) to call \( g \), and function \( g \) to call \( f \)?
Mutual Recursion

Does this work?

```
1 fun f x =  
  2 ... 
  3 g y 
  4 ... 

5 fun g y = 
  6 ... 
  7 f x 
  8 ... 
```
Mutual Recursion

We can employ higher order functions for a work around.

fun earlier (f,x) = ... f y ...
... 
fun later x = ... earlier(later,y)
Mutual Recursion

But ML gives us special syntax for this

1 fun f x = 5 and g y =
2     ... 6    ...
3  g y     7  f x
4     ... 8    ...

Modules

Signatures are good for
• organization and management
• maintaining invariants
Modules - Invariants

Ensuring safe use of data structures
  ◦ e.g. always insert before querying

Keeping data normalized
  ◦ e.g. only store fractions in lowest terms

Enforcing policy
  ◦ e.g. don't allow shipping request without purchase order

Maintain privacy
  ◦ e.g. force authentication for api use
Currying and Higher Order Functions

• `List.map f l`
• `List.filter p l`
• `List.foldl f init l`