# Section 4

CSE 341 – Spencer Pearson

# Mutual Recursion

What if we need function f to call g,
and function g to call f?

# Mutual Recursion

Does this work?

```
    fun f x =
y =
        ...
    ...
        g y
    f x
        ...
    ...
```

```
    fun g
```

# Mutual Recursion

We can employ higher order functions for a work around.

```
fun _earlier f x = … f y …
fun later x = … _earlier later y
val earlier = _earlier later
```

# Mutual Recursion

But ML gives us special syntax for this

```
    fun f x =
y =
        ...
    ...
        g y
    f x
        ...
    ...
```

```
    fun g
```

# Modules

Remember that signatures are good both for organization and management and for maintaining invariants

# Modules - Invariants

- Ordering of operations
  - e.g. insert, then query
- Data kept in good state
  - e.g. fractions in lowest terms
- Policies followed
  - e.g. don't allow shipping request without purchase order

# Currying and Higher Order Functions

Questions?