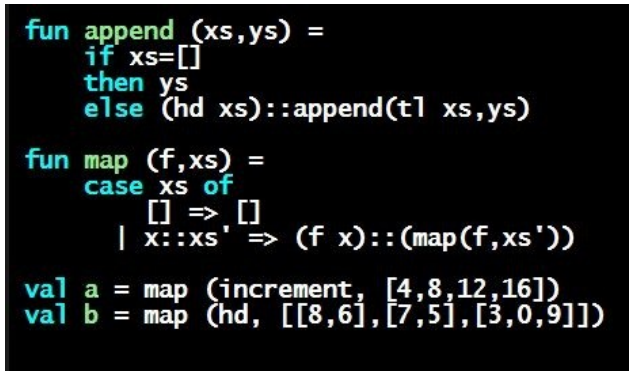

```
fun append (xs,ys) =
    if xs=[]
    then ys
    else (hd xs)::append(tl xs,ys)

fun map (f,xs) =
    case xs of
        [] => []
      | x::xs' => (f x)::(map(f,xs'))

val a = map (increment, [4,8,12,16])
val b = map (hd, [[8,6],[7,5],[3,0,9]])
```

Programming Languages

Dan Grossman
2013

Fold and More Closures

Another famous function: Fold

`fold` (and synonyms / close relatives reduce, inject, etc.) is another very famous iterator over recursive structures

Accumulates an answer by repeatedly applying **`f`** to answer so far

- **`fold(f,acc,[x1,x2,x3,x4])`** computes **`f(f(f(f(acc,x1),x2),x3),x4)`**

```
fun fold (f,acc,xs) =
    case xs of
      []    => acc
    | x::xs => fold(f, f(acc,x), xs)
```

- This version "folds left"; another version "folds right"
- Whether the direction matters depends on **`f`** (often not)

```
val fold = fn : ('a * 'b -> 'a) * 'a * 'b list -> 'a
```

Why iterators again?

- These “iterator-like” functions are not built into the language
  - Just a programming pattern
  - Though many languages have built-in support, which often allows stopping early without resorting to exceptions

- This pattern separates recursive traversal from data processing
  - Can reuse same traversal for different data processing
  - Can reuse same data processing for different data structures
  - In both cases, using common vocabulary concisely communicates intent

Examples with fold

These are useful and do not use "private data"

```
fun f1 xs = fold((fn (x,y) => x+y), 0, xs)
fun f2 xs = fold((fn (x,y) => x andalso y>=0),
                 true, xs)
```

These are useful and do use "private data"

```
fun f3 (xs,hi,lo) =
    fold(fn (x,y) =>
            x + (if y >= lo andalso y <= hi
                 then 1
                 else 0)),
         0, xs)
fun f4 (g,xs) = fold(fn (x,y) => x andalso g y),
                     true, xs)
```

Iterators made better

- Functions like **`map`**, **`filter`**, and **`fold`** are *much* more powerful thanks to closures and lexical scope

- Function passed in can use any "private" data in its environment

- Iterator "doesn't even know the data is there" or what type it has