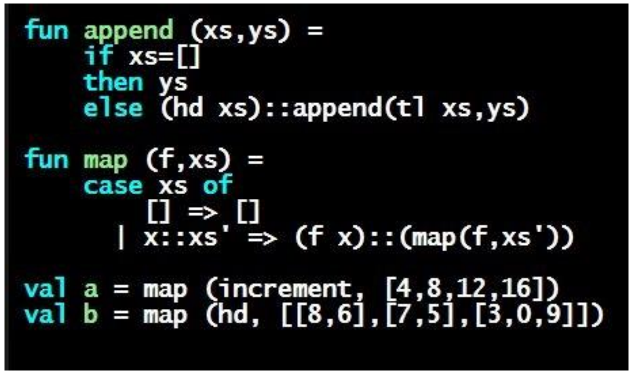
```
fun append (xs,ys) =
    if xs=[]
    then ys
    else (hd xs)::append(tl xs,ys)

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

Type Synonyms

Creating new types

- A *datatype binding* introduces a new type name
  - Distinct from all existing types
  - Only way to create values of the new type is the constructors

- A *type synonym* is a new kind of binding

```
type aname = t
```

  - Just creates another name for a type
  - The type and the name are *interchangeable in every way*
  - Do not worry about what REPL prints: picks what it wants just like it picks the order of record field names

Why have this?

For now, type synonyms just a convenience for talking about types

- Example (where **`suit`** and **`rank`** already defined):

  ```
  type card = suit * rank
  ```

- Write a function of type

  ```
  card -> bool
  ```

- Okay if REPL says your function has type

  ```
  suit * rank -> bool
  ```

Convenient, but does not let us “do” anything new

Later in course will see another use related to modularity