

```
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

Tuples as Syntactic Sugar

# *The truth about tuples*

Previously, we gave tuples syntax, type-checking rules, and evaluation rules

But we could have done this instead:

- Tuple syntax is just a different way to write certain records
- $(e_1, \dots, e_n)$  is another way of writing  $\{1=e_1, \dots, n=e_n\}$
- $t_1 * \dots * t_n$  is another way of writing  $\{1:t_1, \dots, n:t_n\}$
- In other words, records with field names 1, 2, ...

In fact, this is how ML actually defines tuples

- Other than special syntax in programs and printing, they don't exist
- You really can write  $\{1=4, 2=7, 3=9\}$ , but it's bad style

# Syntactic sugar

“Tuples are just **syntactic sugar** for records with fields named 1, 2, ... n”

- *Syntactic*: Can describe the semantics entirely by the corresponding record syntax
- *Sugar*: They make the language sweeter 😊

Will see many more examples of syntactic sugar

- They simplify *understanding* the language
- They simplify *implementing* the language

Why? Because there are fewer semantics to worry about even though we have the syntactic convenience of tuples

Another example we saw: **andalso** and **orelse** vs. **if then else**