```
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

Shadowing

Multiple bindings of same variable

Multiple variable bindings of the same variable is often poor style

- Often confusing

But it's an instructive exercise

- Helps explain how the environment "works"
- Helps explain how a variable binding "works"

(Emphasize this now to lay the foundation for first-class functions)

Our example

```
val a = 10

val b = a * 2

val a = 5

val c = b

val d = a

val a = a + 1

(* val g = f - 3 *)(* does not type-check *)

val f = a * 2
```

Two reasons (either one sufficient)

```
val a = 1
val b = a (* b is bound to 1 *)
val a = 2
```

1. Expressions in variable bindings are evaluated “eagerly”
   - Before the variable binding “finishes”
   - Afterwards, the expression producing the value is irrelevant

2. There is no way to “assign to” a variable in ML
   - Can only shadow it in a later environment

use

This is why I am so insistent about not reusing **`use`** on a file without restarting the REPL

Else you are introducing some of the same bindings again

- May make it seem like wrong code is correct
- May make it seem like correct code is wrong
- (It's all well-defined, but we humans get confused)