

```
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

Optional: Closure Idioms Without Closures

Higher-order programming

- Higher-order programming, e.g., with **map** and **filter**, is great
- Language support for closures makes it very pleasant
- Without closures, we can still do it more manually / clumsily
 - In OOP (e.g., Java) with one-method interfaces
 - In procedural (e.g., C) with explicit environment arguments
- Working through this:
 - Shows connections between languages and features
 - Can help you understand closures and objects

Outline

This segment:

- Just the code we will “port” to Java and/or C
- Not using standard library to provide fuller comparison

Next segments:

- The code in Java and/or C
- What works well and what is painful