

```
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

*The Pieces of Learning a Language*

# *Five different things*

1. **Syntax:** How do you write language constructs?
2. **Semantics:** What do programs mean? (Evaluation rules)
3. **Idioms:** What are typical patterns for using language features to express your computation?
4. **Libraries:** What facilities does the language (or a well-known project) provide “standard”? (E.g., file access, data structures)
5. **Tools:** What do language implementations provide to make your job easier? (E.g., REPL, debugger, code formatter, ...)
  - Not actually part of the language

These are 5 separate issues

- In practice, all are essential for good programmers
- Many people confuse them, but shouldn't

# *Our Focus*

This course focuses on semantics and idioms

- Syntax is usually uninteresting
  - A fact to learn, like “The American Civil War ended in 1865”
  - People obsess over subjective preferences
- Libraries and tools crucial, but often learn new ones “on the job”
  - We are learning semantics and how to use that knowledge to understand all software and employ appropriate idioms
  - By avoiding most libraries/tools, our languages may look “silly” but so would *any* language used this way