

```
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

Introduction to Racket

Racket

Next two sections will use the Racket language (not ML) and the DrRacket programming environment (not Emacs)

- Installation / basic usage instructions on course website
- Like ML, functional focus with imperative features
 - Anonymous functions, closures, no return statement, etc.
 - But we will not use pattern-matching
- Unlike ML, no static type system: accepts more programs, but most errors do not occur until run-time
- Really minimalist syntax
- Advanced features like macros, modules, quoting/eval, continuations, contracts, ...
 - Will do only a couple of these

Racket vs. Scheme

- Scheme and Racket are very similar languages
 - Racket “changed its name” in 2010
 - Please excuse any mistakes when I speak
- Racket made some non-backward-compatible changes...
 - How the empty list is written
 - Cons cells not mutable
 - How modules work
 - Etc.... and many additions
- Result: A modern language used to build some real systems
 - More of a moving target (notes may become outdated)
 - Online documentation, particularly “The Racket Guide”

Getting started

DrRacket “definitions window” and “interactions window” very similar to how we used Emacs and a REPL, but more user-friendly

- DrRacket has always focused on good-for-teaching
- See usage notes for how to use REPL, testing files, etc.
- Easy to learn to use on your own, but lecture demos will help

Free, well-written documentation:

- <http://racket-lang.org/>
- The Racket Guide especially,
<http://docs.racket-lang.org/guide/index.html>

File structure

Start every file with a line containing only

#lang racket

(Can have comments before this, but not code)

A file is a module containing a *collection of definitions* (bindings)...