



#### CSE341: Programming Languages

#### Lecture 26 Course Victory Lap

Dan Grossman Spring 2013

# Final Exam

٠

As also indicated in class-list email:

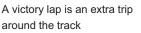
- Next Thursday, 8:30-10:20
  - Intention is to focus primarily on material since the midterm - Including topics on homeworks and not on homeworks
  - May also have a little ML, just like the course has had
- · You will need to write code and English
- · I hope you will pick up your exams when available Including beginning of Fall Quarter

Spring	2013		

CSE341: Programming Languages

2

### Victory Lap



By the exhausted victors (us) ☺

Review course goals

- Slides from Introduction and Course-Motivation

Some big themes and perspectives

- Stuff for five years from now more than for the final

Course evaluations: please do take some time

Spring 2013

CSE341: Programming Languages

# Thank you!

- Huge thank-you to your TAs
  - Great team effort
  - Deep understanding of material despite all having different 341 instructors
  - Great sections, timely grading, etc., etc.

Spring 2013

Spring 2013

[From Lecture 1]

CSE341: Programming Languages

· Many essential concepts relevant in any programming language

- Using multiple languages shows how the same concept can

4

# Thank you!

- And a huge thank you to all of you
  - Great attitude about a very different view of software
  - Good class attendance and questions
  - Occasionally laughed at stuff ©
- Computer science ought to be challenging and fun!

3

- But many other topics too

- And how these pieces fit together

- They let many of the concepts "shine"

"look different" or actually be slightly different

- Not using mutation (assignment statements) (!) - Using first-class functions (can't explain that yet)

· Use ML, Racket, and Ruby languages:

- In many ways simpler than Java · Big focus on functional programming

6

## [From Lecture 1]

Learning to think about software in this "PL" way will make you a better programmer even if/when you go back to old ways

It will also give you the mental tools and experience you need for a lifetime of confidently picking up new languages and ideas

[Somewhat in the style of The Karate Kid movies (1984, 2010)]



Spring 2013

CSE341: Programming Languages

### [From Course Motivation]

SML, Racket, and Ruby are a useful combination for us

	dyn	namically typ	bed	statically t	yped	
functional		Racket		SML		
object-orient	ed	Ruby		Java		
a characteristic transport and the up and the indicate of the set to the set of the set						

ML: polymorphic types, pattern-matching, abstract types & modules Racket: dynamic typing, "good" macros, minimalist syntax, eval Ruby: classes but not types, very OOP, mixins [and much more]

Really wish we had more time: Haskell: laziness, purity, type classes, monads Prolog: unification and backtracking [and much more]

Spring 2013

CSE341: Programming Languages

#### Some other highlights

- Function closures are *really* powerful and convenient...
   ... and implementing them is not magic
- Datatypes and pattern-matching are really convenient...
   ... and exactly the opposite of OOP decomposition
- Sound static typing prevents certain errors...
   ... and is inherently approximate
- Subtyping and generics allow different kinds of code reuse...
   ... and combine synergistically
- · Modularity is really important; languages can help

Spring 2013

CSE341: Programming Languages

#### 11

9

#### [From Course Motivation]

- No such thing as a "best" PL
- Fundamental concepts easier to teach in some (multiple) PLs
- A good PL is a relevant, elegant interface for writing software
  There is no substitute for precise understanding of PL semantics
- Functional languages have been on the leading edge for decades
  - Ideas have been absorbed by the mainstream, but very slowly
  - First-class functions and avoiding mutation increasingly essential
  - Meanwhile, use the ideas to be a better C/Java/PHP hacker
- Many great alternatives to ML, Racket, and Ruby, but each was chosen for a reason and for how they complement each other

CSE341: Programming Languages

8

#### Benefits of No Mutation

#### [An incomplete list]

Spring 2013

- 1. Can freely alias or copy values/objects: Unit 1
- 2. More functions/modules are equivalent: Unit 4
- 3. No need to make local copies of data: Unit 5
- 4. Depth subtyping is sound: Unit 8

State updates are appropriate when you are modeling a phenomenon that is inherently state-based

– A fold over a collection (e.g., summing a list) is not!

Spring 2013

CSE341: Programming Languages

## From the syllabus

Successful course participants will:

- Internalize an accurate understanding of what functional and object-oriented programs mean
- Develop the skills necessary to learn new programming languages quickly
- Master specific language concepts such that they can recognize them in strange guises
- Learn to evaluate the power and elegance of programming languages and their constructs
- Attain reasonable proficiency in the ML, Racket, and Ruby languages and, as a by-product, become more proficient in languages they already know

Spring 2013

# The End

This really is my favorite course and it probably always will be



Don't be a stranger!

Spring 2013

CSE341: Programming Languages

13