Introduction to COS441

Princeton University
Fall 2004

Teaching Assistant

Frances Spalding
frances@cs.princeton.edu
Room 316
Office Hours
Monday 1:30 – 2:30
Tuesday 3:30 – 4:30

Course Content

- Mathematical Foundations
  - Syntax, Semantics, Type Soundness ...
- Using Advanced PL features
  - Type systems, objects, advanced control flow
- Implementation techniques
  - Evaluators, Abstract Machines, DSL,…

The Class

Lecturer

Daniel Wang
danwang@cs.princeton.edu
Room 205
Office Hours
Monday and Friday
11:00-12:00

Pictures

After class let me snap your picture if you're not shown above!

Send me email if you prefer to be addressed by a different name

Send me email if you're not on the course list!

Logistics

- Make sure you have a CS account
  - See course web for getting started with SML
- First assignment handed out Wednesday
  - Due one week later
- Feel free to set up an appointments if the offices hours don't work for you

Work

- Written exercises taken from the reading
  - Mainly just to improve reading comprehension
- Semi-formal proofs
  - Based on lecture and reading
- Weekly Programming Assignments
  - Mostly in ML, a few in Java

Grading

50% 9 assignments

20% take-home midterm

30% take-home final

See web page for collaboration policy

Graduate Students on a separate curve

Extra credit factored in after computing the curve

Schedule

- See course web for schedule
  - Will make slides available after each lecture
- Assignments passed out Wednesday
  - Usually due back one week later
- Do reading listed before coming to class
  - Lectures will cover material in the reading and things related to but not in reading

Reading

- Course Notes from Bob Harper
  - A bit too terse in some places
  - Lectures will walk you through them
- Mitchell - "Concepts in Programming Languages"
  - An easier read with interesting historical facts and a more informal approach
- Ullman - "Elements of ML Programming"
  - Reference guide to ML programming

Very Rough Outline of the Course

- 1st half is basic theoretical foundations of programming languages
  - Proofs and programming
- 2nd half is more about applications and applied theory
  - Fewer proofs more programming
  - Discussion of advanced programming techniques and features

Programming Assignments

- Will try to emphasize practical implementation techniques for interpreters
- Basic techniques are independent of programming language
  - Will do some of this in Java and ML

Why Study PLs?

- A programming language is the basic building material used to build software
- Studying a PL is like studying new mixtures of concrete!
- PLs are just as exciting concrete!

Written Assignments

- Try to emphasize the foundations behind intuitions
  - When you say this is "obviously so" one ought to have a proof sketch in your head
- Inductive/recursive reasoning
  - All you need to know about programming languages and problem solving is to really understand induction

Why Study PLs?

- A programming language is the basic building material used to build software
- Studying a PL is like studying new mixtures of concrete!
- PLs are just as exciting concrete!

Concrete is really exciting

Well… okay… you need some convincing don't you

Motivations

and other crazy ramblings

Why Concrete is Exciting

- The Roman empire was built with concrete!
- Easy to mold and with many uses
- Made up of a filler and binder
  - Mixing different fillers and binders you can vary the properties of the concrete
- Spend a day identifying all the things you rely on using concrete
  - BTW Asphalt is a form of concrete

Questions about Concrete

- What are the basic physics and chemistry of concrete?
- What new kind of concretes can we make with different fillers and binders?
- What can we build with different kinds of concrete?

Why PLs are Exciting

- Microsoft's empire is built with PLs
  - Java barbarians are trying to take over
- PLs are amazingly flexible
- A PL is a library of primitives plus ways of gluing them together
  - We can make different PLs by varying the primitives or the glue
- Just think about how reliant we now are on PLs

Roman Pantheon

2000 year old concrete dome

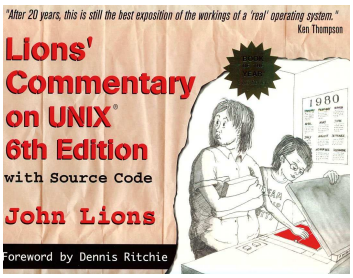
Questions about PLs

- What are the mathematics principles behind PLs?
- What kind of new PLs can we make by mixing primitives and gluing them together in different ways?
- What new software architectures can be built using novel PLs?

The Nassau Hull

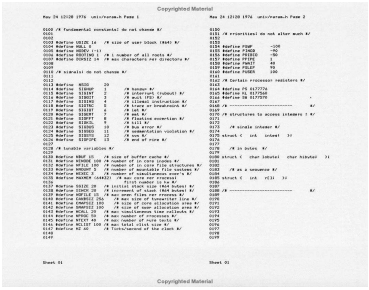
Princeton's 2002
National Concrete Canoe Competition Entry

Unix



almost 30 year old OS

Unix

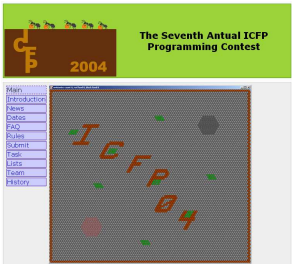


almost 30 year old OS

Learning to programming in a functional programming language is like building a concrete canoe

You learn lots of amazing things!

ICFP Programming Contest



Learning to programming in a functional programming language is like building a concrete canoe

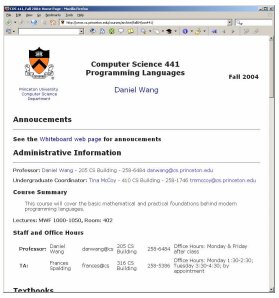
You learn lots of amazing things!

Even if at first it doesn't sound practical.

Learning to programming in a functional programming language is like building a concrete canoe

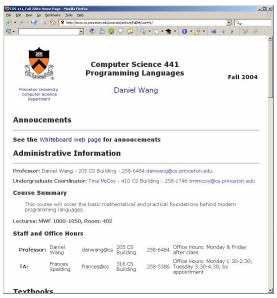
Question for The Class

What languages were involved in making of this web page?



Question for The Class

1. C
2. HTML
3. RDF
4. Java
5. JavaScript
6. PHP
7. sh
8. ??
9. XML/Perl
10. C++
11. Python
12. awk
13. latex
14. COBOL
15. Lex/Yacc
16. XUL
17. SML
18. Tcl
19. CSS
20. ELISP
21. FORTRAN
22. VB

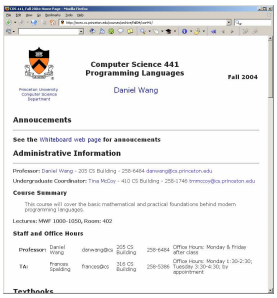


The "Fastest" PL

- Which language is the "fastest"
- Fastest at what?
- Fastest in terms of CPU time, programmer time, or time to market?
- To understand or critique a programming language there must be some context
  - We must compare it to available alternatives

Question for The Class

Which of those languages is "best" or "fastest"?



The "Fastest" PL

- Which language is the "fastest"
- Fastest at what?
- Fastest in terms of CPU time, programmer time, or time to market?
- To understand or critique a programming language there must be some context
  - We must compare it to available alternatives
- But if you want to talk about bogus benchmarks!

The "Best" PL

- There is no "best" PL
- There may be a best PL for a particular job
  - There may be several "best" PLs too
- There may be no good PL at all
  - Time to invent a new one
- ML is great for teaching programming language concepts among other things
  - So that's why we will use it in this course for that purpose

Programming Language Shootout

http://shootout.alioth.debian.org/craps.php

Completely Random Arbitrary Point System

SCORES

Language	Implementation	Score	Missing
C	gcc	51.6925	0
C++	g++	40.5100	0
C#	csharp	39.9038	0
SML	mlton	39.2517	0
Eiffel	se	37.7553	3
Ocaml	ocaml	36.2406	0

Programming Language Shootout

[Shootout-list] Re: Timing issues

Isaac Gouy igouy2@yahoo.com
Sun, 5 Sep 2004 11:00:24 -0700 (PDT)

- Previous message: [Shootout-list] Re: Timing issues
- Next message: [Shootout-list] Re: Timing issues
- **Messages sorted by:** [date] [thread] [subject] [author]

```
> I wrote to the kernel developers: it looks like timing info is not
> reported from subthreads unless the subthread exits: it's a timing
> issue. Sometimes after executing the C# program the subthread that
> executes it manages to call pthread_exit() before the main program
> quits and sometimes not, so you get the incorrect timings.
>
> This could be considered a buglet in mono, though the time results
> are the only thing that should be affected by it. Maybe Dick has
> further insights why we don't properly wait for a thread with
> pthread_join() (ie why we use pthread_detach(): there may be other
> ways for us to wait for the thread to call pthread_exit ()).
>
> Lupus
```

Domain Specific Languages

- Very likely few of anyone in this class will be designing the next Java or C#
- However, very likely you will find yourself in a situation where the right thing to do is to invent a specialized language
- Domain Specific Languages
  - They are all over the place

Programming Language Shootout

http://shootout.alioth.debian.org/craps.php

Completely Random Arbitrary Point System

SCORES

Language	Implementation	Score	Missing
C	gcc	51.6925	0
C++	g++	40.5100	0
SML	mlton	39.2517	0
Eiffel	se	37.7553	3
Ocaml	ocaml	36.2406	0

Some DSLs

SQL, Maple, Mathematica Matlab, S-PLUS, SPSS, SAS, PHP, Postscript, PDF, PHP, ActionScript, Perl, awk, sed, XHTML, XML, XQuery, OpenGL Shader Language, CG, ….

Appreciating the Finer Points

- It's hard to evaluate the design or a PL if you don't understand the technical limits of the "state of the art"
- Any idiot can make a bridge stand up, but it takes a engineer to make it just **barely** stand up
- Just be cause it "works" doesn't mean its designed right and just isn't over engineered

Summary

- There are lots of interesting little languages being used and invented
  - You might be responsible for one in the future
  - Or stuck maintaining someone else!
- We've learned about how to think about PLs in a formal mathematical way
  - The rigor is helps you identify design problems with earlier rather than later
  - There will be a little more rigor than most people are used to in this class (trust me it's good for you!)

Reminder

- Pequod claims they have reprinted the course notes
- When you pick them up make sure they printed every page not just the even ones!
- Harper's notes are online too
  - Read them their if you can't get the notes from Peqoud
  - Read Chapter 1 of Harper's notes before Monday