Course Introduction

Dr. Mattox Beckman

University of Illinois at Urbana-Champaign
Department of Computer Science

Welcome to CS 421!

Topics for discussion:

- Logisitics – instructor, course objectives
- Why study languages?
- Major themes for the course

Me!

Name	Mattox Beckman
History	PhD, Fall 2003, University of Illinois at Urbana-Champaign Lecturer 2003–2015 Illinois Institute of Technology
Research Areas	CS Education, Programming Languages, Mathematical Foundations of Computer Science
Specialty	Partial Evaluation, Functional Programming
Professional Interests	Teaching; Computer Science Education; Functional Programming; Semantics and Types; Category Theory
Personal Interests	Irish Music; Cooking; Go (Baduk, Wei-Qi, Igo); Philosophy; Evolution; Meditation; Kerbal Space Program; Home-brewing; ... and many many more ...

Activities

- This is a **flipped** classroom!
  - Please watch the lecture video *before* coming to class!
- In class activities to reinforce learning.
- Prairielearn post-class activities to consolidate/apply learning.
- There is not necessarily a post-class activity for each day.

Machine Problems

- Designed to help you study for the exams, and to achieve major course objectives
- You are allowed one partner for the programming part, but **you must cite your sources!** (Place partner netids in a comment at the top.)
- Don't use the "perturbation method" of solving machine problems! We expect you to *understand* the solution and the process very well.
- See the syllabus for more details.

Exams/Quizzes

- The purpose of an exam is to measure mastery of material.
  - Exams are subdivided into proficiency units.
  - The final exam will retest many of the proficiency units. If you improve your score, we update your midterm score with it!
- Three midterms
- Final exam: “Second Chance”

Why Study Languages?

- *Pai sei*
- Blub – see *Beating the Averages* by Paul Graham. [Gra03]
- Language families

Pai Sei

Different languages can express different concepts efficiently!

- A story from human languages: *pai sei*

Pai Sei

Different languages can express different concepts efficiently!

- A story from human languages: *pai sei*
- Languages and cultures grow together to shape each other.

Pai Sei

Different languages can express different concepts efficiently!

- A story from human languages: *pai sei*
- Languages and cultures grow together to shape each other.
- It's difficult to reason about something without vocabulary!

Pai Sei

Different languages can express different concepts efficiently!

- A story from human languages: *pai sei*
- Languages and cultures grow together to shape each other.
- It's difficult to reason about something without vocabulary!
- See *Politics and the English Language* by George Orwell. [Orw46]

Blubs

- From *Beating the Averages* by Paul Graham

Blubs

- From *Beating the Averages* by Paul Graham
- The difference between a known powerful language to a less powerful language is easy to see.

Blubs

- From *Beating the Averages* by Paul Graham
- The difference between a known powerful language to a less powerful language is easy to see.
- The difference between a known less powerful language to a more powerful language is not easy to see!

Themes

The course has four major parts:

1. Functional Programming
   You will learn functional programming by learning how to build interpreters in HASKELL.

Themes

The course has four major parts:

1. Functional Programming
   You will learn functional programming by learning how to build interpreters in HASKELL.
2. Parsing
   You will learn how text becomes a data structure we can use to represent a program.

Themes

The course has four major parts:

1. Functional Programming
   You will learn functional programming by learning how to build interpreters in HASKELL.
2. Parsing
   You will learn how text becomes a data structure we can use to represent a program.
3. Mathematical Foundations
   You will learn some of the mathematical theory that lets us reason about programming languages and the programs written in them.

Themes

The course has four major parts:

1. Functional Programming
   You will learn functional programming by learning how to build interpreters in HASKELL.
2. Parsing
   You will learn how text becomes a data structure we can use to represent a program.
3. Mathematical Foundations
   You will learn some of the mathematical theory that lets us reason about programming languages and the programs written in them.
4. Pragmatics
   You will learn some of the design decisions available to you when choosing (or creating!) a language.

So, what should you learn?

- Understand major classes of programming languages: techniques, features, styles.
- How to select an appropriate language for a given task.
- How to read a formal specification of a language and implement it.
- How to write a formal specification of a language.
- Some Powerful Ideas:
  1. Recursion
  2. Abstraction
  3. Transformation
  4. Unification

The emphasis is on learning the theory, knowing why the theory is valuable, and using it to implement a language.

Project Euler

- Multiples of 3 and 5: If we list all the natural numbers below 10 that are multiples of 3 or 5, we get 3, 5, 6 and 9. The sum of these multiples is 23. Find the sum of all the multiples of 3 or 5 below 1000.
- The prime factors of 13195 are 5, 7, 13 and 29. What is the largest prime factor of the number 600851475143?

Bibliography

[Bac97] John Backus. "Can Programming Be Liberated from the von Neumann Style? A functional Style and Its Algebra of Programs." In: ACM Turing Award Lecture (1997).

[Gra03] Paul Graham. *Beating the Averages*. Apr. 2003. URL: http://www.paulgraham.com/avg.html.

[Orw46] George Orwell. "Politics and the English Language." In: *Horizon* 13.76 (Apr. 1946), pp. 252–265. URL: http://www.resort.com/~prime8/Orwell/patee.html.