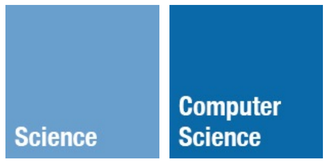
Preliminaries



CS 440: Programming Languages
Michael Lee <lee@iit.edu>

Michael Lee

- lee@iit.edu
- http://moss.cs.iit.edu
- Office: SB 226C
- Hours: Tue/Thu 9:30AM-12:30PM on Zoom
  (make appointment on my homepage)

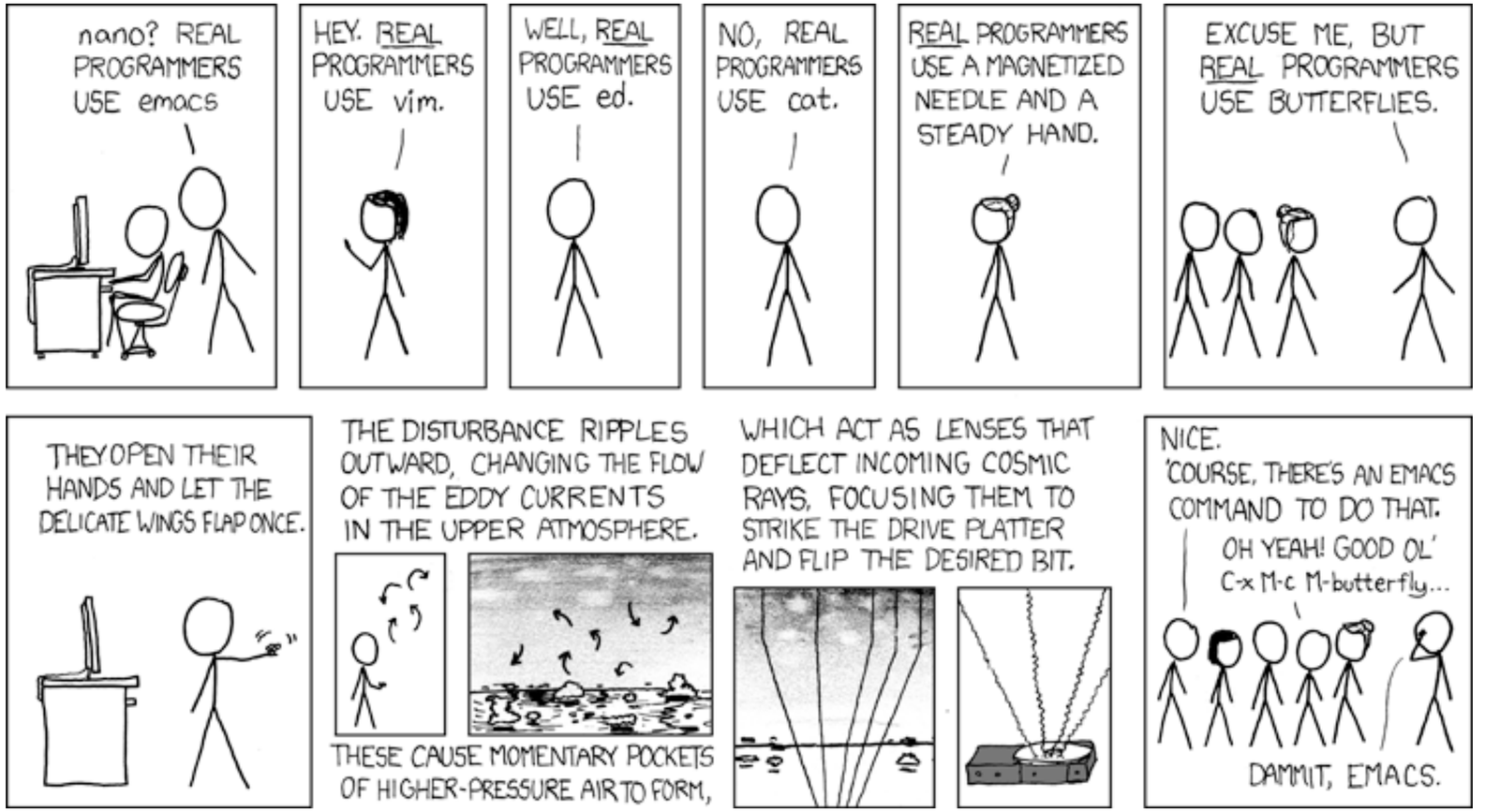
TA: Xincheng Yang

- xyang76@hawk.iit.edu
- Hours: Tue/Thu 3PM-4PM or by appointment

Agenda

- Course overview
- Administrivia
  - Grading
  - Assessments
  - Resources

§ Programming Languages


nano? REAL PROGRAMMERS USE emacs
HEY. REAL PROGRAMMERS USE vim.
WELL, REAL PROGRAMMERS USE ed.
NO, REAL PROGRAMMERS USE cat.
REAL PROGRAMMERS USE A MAGNETIZED NEEDLE AND A STEADY HAND.
EXCUSE ME, BUT REAL PROGRAMMERS USE BUTTERFLIES.
THEY OPEN THEIR HANDS AND LET THE DELICATE WINGS FLAP ONCE.
THE DISTURBANCE RIPPLES OUTWARD, CHANGING THE FLOW OF THE EDDY CURRENTS IN THE UPPER ATMOSPHERE.
THESE CAUSE MOMENTARY POCKETS OF HIGHER-PRESSURE AIR TO FORM,
WHICH ACT AS LENSES THAT DEFLECT INCOMING COSMIC RAYS, FOCUSING THEM TO STRIKE THE DRIVE PLATTER AND FLIP THE DESIRED BIT.
NICE.
'COURSE, THERE'S AN EMACS COMMAND TO DO THAT.
OH YEAH! GOOD OL' C-x M-c M-butterfly...
DAMMIT, EMACS.

Prereqs

- Programming experience (likely imperative & OOP)
- Some architecture knowledge
- Analysis of algorithms
- Notion of language equivalence?

Programming Languages …

- Are theoretically all the same, but yet practically very different!

Not just a consumer!

- Dissect, categorize, analyze, and reassemble languages
- Learn to modify and create your own languages

We will …

1. Use a new language, Racket, to learn about different programming language constructs and ideas.
2. Learn about different methods of language specification, focusing on *semantics* and *verification*.
3. Analyze how programs are *interpreted*, *compiled*, *represented*, *evaluated*, and *optimized*.
4. Implement interpreters for a handful of different languages.

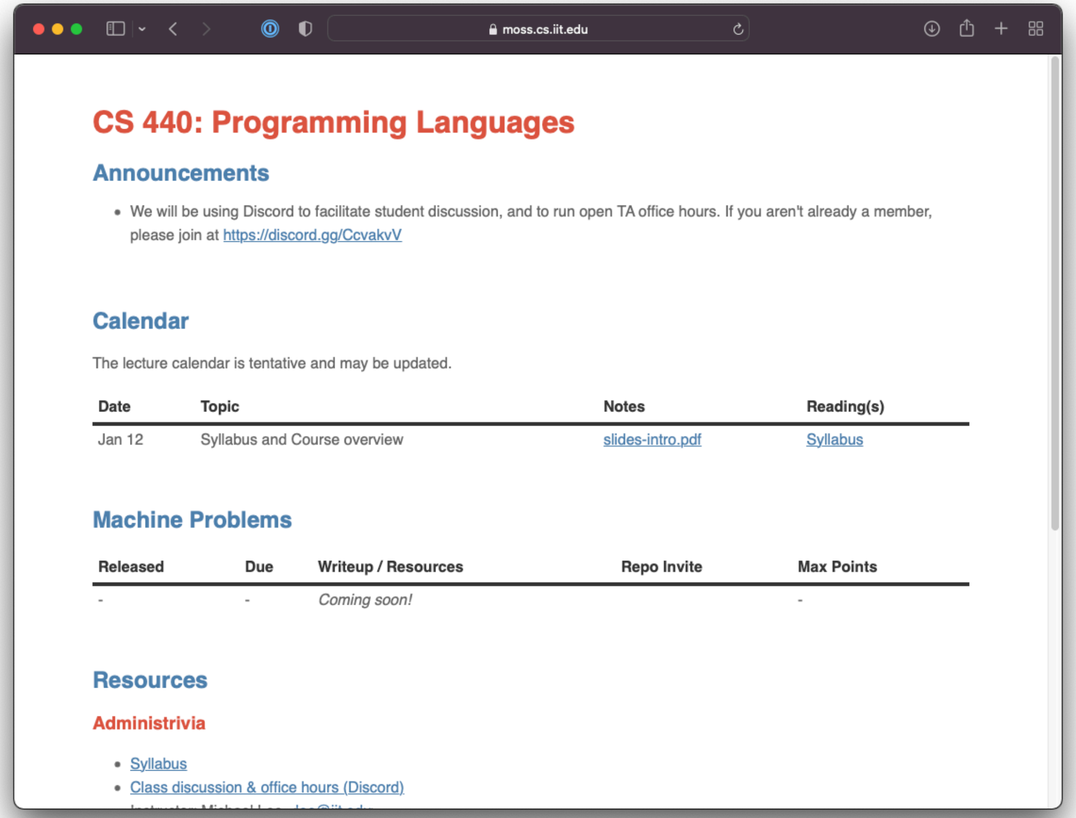
Topics

- Racket
- Higher order functions
- Recursion
- Closures
- Metaprogramming
- Syntax
- Parsing
- Grammars and Languages
- Semantics
- Evaluation strategies
- Interpreters
- Operational semantics
- Type inference and Unification
- Memory management

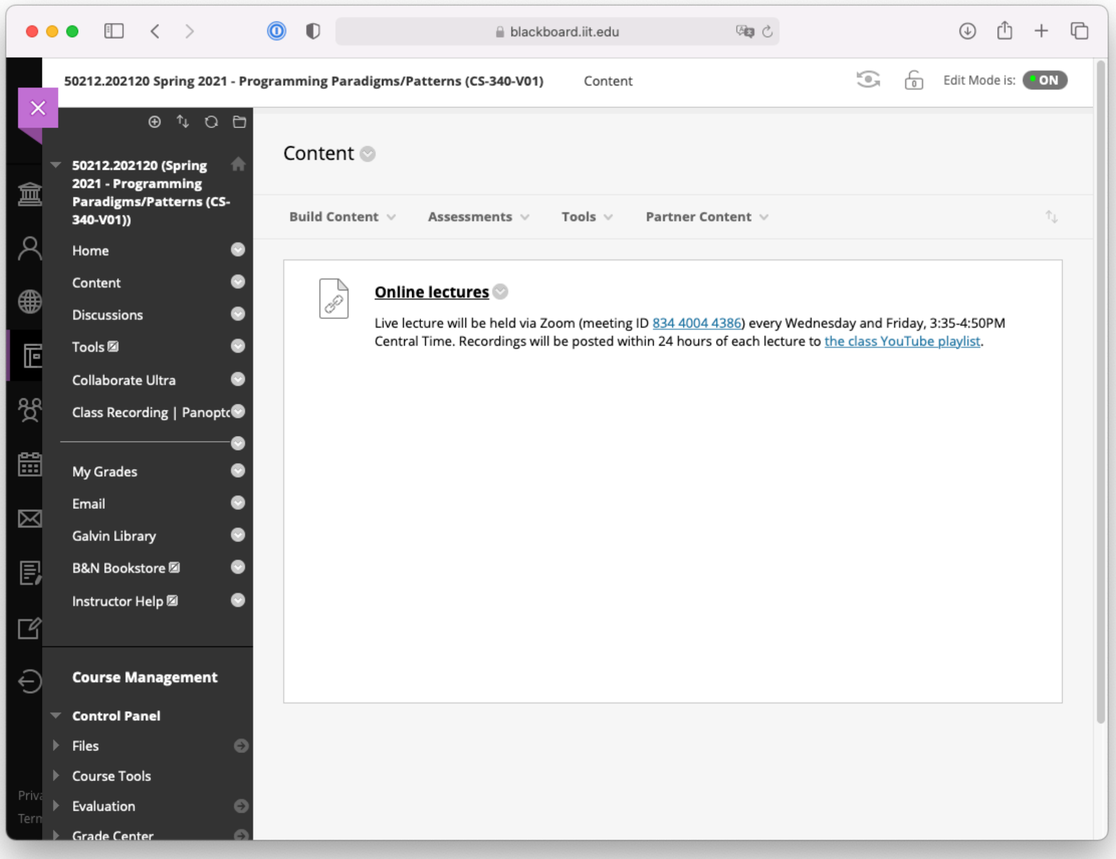
§ Administrivia

Prerequisites

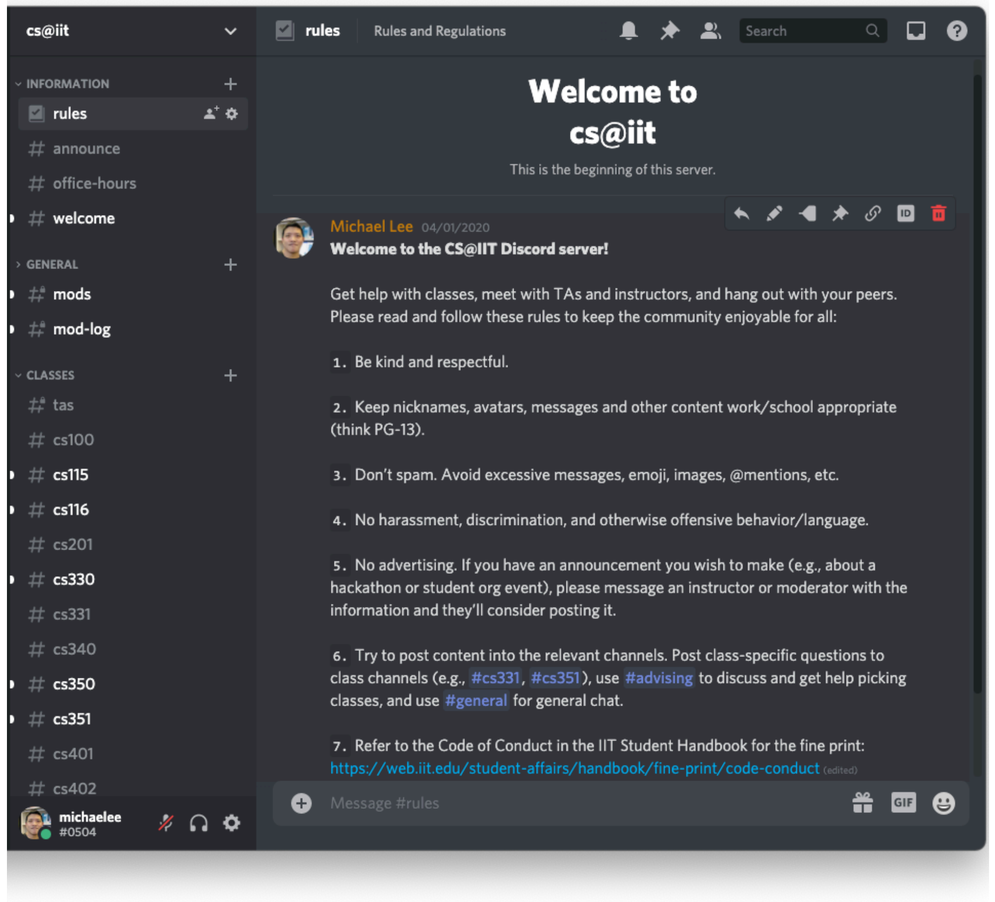
- I assume you are …
  - fluent in some programming language
    - familiar with procedural & OO paradigms
  - comfortable with development processes:
    - compilation, debugging, testing



Course website: http://moss.cs.iit.edu/cs440



Blackboard: http://blackboard.iit.edu



Discord: TA class discussion and Q/A (invite on course website)

References

- Programming Languages: Application and Interpretation, by Shriram Krishnamurthi
- Crafting Interpreters, by Robert Nystrom
- Compilers: Principles, Techniques, & Tools, 2nd edition, by Aho, Lam, Sethi & Ullman, 2007.

Grading

- 60% Assignments
- 20% Midterm Exam
- 20% Final Exam (Cumulative)

Assignments

- 6-8 total
  - Some written, some machine problems (coding problems)
    - Written submitted via Blackboard, MPs via GitHub

Late Policy

- 7-day late pool, distributed however you like across labs (a day at a time)
- If you're out of late days, late submissions will not be accepted!

Exams

- Midterm and Final exams both administered online, both open-book, open-notes
- Scores may be linearly scaled so that median/mean (whichever lower) is 75%
- Midterm tentatively scheduled for **March 4**

A: ≥ 90%

B: 80-89%

C: 70-79%

D: 60-69%

E: < 60%

For Friday

- Read chapter 2 of *Crafting Interpreters*: "A Map of the Territory"