End-of-semester logistics

- Office hours 2-3pm **on Zoom** (same as Monday link)
- HW5 Due tonight (Saturday with 2 late days)
- Late submissions not accepted – I need to post solutions

- Zoom office hours Monday
  - Come with any questions ahead of the final

- **Final:** Wednesday, May 1, 2:00-4:00pm, SB 238
- I’m assuming everyone is taking it in person unless I’ve heard from you already
Final Exam

• All homeworks, all lectures (including lectures since HW5)
• **Three (3)** letter-size (8.5x11”) double-sided sheets of notes
• I’ll provide reference material (posted ahead of time on BB)

Tentative format:

• Short answer
• Type -> Expression, Expression -> Type
• 3-5 longer questions (like midterm, HW)
Today: Equational Reasoning

\((\lambda x. e) \nu\) vs \([e/x] \nu\)

Corporate needs you to find the differences between this picture and this picture.

They're the same picture.
Course Objectives

• Understand the models, notations and techniques used by Programming Languages researchers to formalize languages and systems.
• Formally prove properties about programming languages and programs.
• Use the above techniques to design new type systems and models to formalize and prove properties about new and existing language features and systems.
• Read and understand professional literature in the field of Programming Languages.
Understand the models, notations and techniques used by Programming Languages researchers to formalize languages and systems

• Inference Rules
• Rule Induction
• Language models
  • IMP
  • Lambda calculus
  • STLC
  • System F
• Dynamic Semantics
  • Small-step
  • Big-step
  • Evaluation contexts

• Control stacks
• Parallel semantics
• Static Semantics
  • Polymorphism
  • Recursive types
  • Subtyping
  • Dependent types
• Curry-Howard
• Equational Reasoning
Formally prove properties about programming languages and programs

• Type Safety
  • Progress
  • Preservation

• Memory Safety

• Security (noninterference)

• Resource usage

• Parametricity
Use the above techniques to design new type systems and models to formalize and prove properties about new and existing language features and systems

• Model garbage collection
• Type system for noninterference
• Static array bounds checking with dependent types
• Type system for amortized analysis
Read and understand professional literature in the field of Programming Languages

• Uh, well... try it (or doing research yourself!)

One recipe for a PL paper:

• Identify a problem (like out-of-bounds array accesses, resource usage, memory safety) you want to model or solve with PL techniques

• Sufficiently formalize the model

• Prove properties of interest