Foundations: Lambda calculus

- Original intention
  - Study foundations of mathematics (Church & Kleene)
- More successful for computable functions
  - Substitution --> symbolic computation
  - Church/Turing thesis
- Influenced design of Lisp, ML, other languages
  - See Boost Lambda Library for C++ function objects
- Important part of CS history and foundations

Lambda calculus

- Formal system with three parts
  - Notation for function expressions (syntax)
  - Proof system for equations
  - Calculation rules called reduction
- Additional topics in lambda calculus
  - Mathematical semantics (=model theory)
  - Type systems

We will look at syntax, equations and reduction

There is more detail in the book than we will cover in class

Why study lambda calculus now?

- Simple, uniform notation for basic PL concepts
  - Free and bound variables
  - Functions and application
  - Declarations
- Calculation rule
  - Symbolic evaluation useful for discussing programs
  - Used in optimization (in-lining), macro expansion
    - ◊ Correct macro processing requires variable renaming
  - Illustrates some ideas about scope of binding
    - ◊ Lisp originally departed from standard lambda calculus, returned to the fold through Scheme, Common Lisp