Untyped Lambda Calculus

Structure and Evaluation, Currying, Church Encodings

Sven Tennie

August 8, 2018

Dream IT
https://dreamit.de

Introduction

Lambda Calculus

- Invented by Alonzo Church (1920s)
- Equally expressive to the Turing Machine(s)
- Formal Language
- Computational Model
    - Lisp (1950s)
    - ML
    - Haskell
- "Lambda Expressions" in almost every modern programming language

Why should I care?

- Simple Computational Model
    - to describe structure and behaviour (E.g. Operational Semantics, Type Systems)
    - to reason and prove

Why should I care?

- Simple Computational Model
  - to describe structure and behaviour (E.g. Operational Semantics, Type Systems)
  - to reason and prove
- Explains why things in FP are like they are
  - Pure Functions
  - Higher-Order Functions
  - Currying
  - Lazy Evaluation

Why should I care?

- Simple Computational Model
  - to describe structure and behaviour (E.g. Operational Semantics, Type Systems)
  - to reason and prove
- Explains why things in FP are like they are
  - Pure Functions
  - Higher-Order Functions
  - Currying
  - Lazy Evaluation
- Understand FP Compilers
  - Introduce FP stuff into other languages
  - Write your own compiler
  - GHC uses an enriched Lambda Calculus internally

Basics

Untyped Lambda Calculus

$t ::=$	Terms:
$\quad x$	Variable
$\quad \lambda x.t$	Abstraction
$\quad t\ t$	Application

Untyped Lambda Calculus

$$
\begin{array}{llr}
t ::= & & \text{Terms:} \\
 & x & \text{Variable} \\
 & \lambda x.t & \text{Abstraction} \\
 & t\ t & \text{Application}
\end{array}
$$

Example - Identity

Lambda Calculus

$$\underbrace{\underbrace{\lambda x.x}_{\text{Abstraction}} \quad \underbrace{y}_{\text{Variable}}}_{\text{Application}} \rightarrow y$$

Untyped Lambda Calculus

$t ::=$	Terms:
x	Variable
$\lambda x.t$	Abstraction
$t\ t$	Application

Example - Identity

Lambda Calculus

$$\underbrace{\underbrace{\lambda x.x}_{\text{Abstraction}} \quad \underbrace{y}_{\text{Variable}}}_{\text{Application}} \to y$$

Javascript

$$\underbrace{(\underbrace{\mathit{function}\ (x)\{\mathit{return}\ x;\}}_{\mathit{Abstraction}})\ (\ \underbrace{y}_{\mathit{Variable}}\ )}_{\mathit{Application}}$$

Example - $(\lambda x.\lambda y.x\ y)\ a\ b$

Abstractions
Think: Function Definitions

$(\underline{\lambda x.\underline{\lambda y.x\ y}})\ a\ b$

Example - $(\lambda x.\lambda y.x\ y)\ a\ b$

Abstractions
Think: Function Definitions

$(\underline{\lambda x.\underline{\lambda y.x\ y}})\ a\ b$

Variables
Think: Parameters

$(\lambda x.\lambda y.\underline{x}\ \underline{y})\ \underline{a}\ \underline{b}$

Abstractions
Think: Function Definitions

$(\underline{\lambda x.\underline{\lambda y.x\ y}})\ a\ b$

Variables
Think: Parameters

$(\lambda x.\lambda y.\underline{x}\ \underline{y})\ \underline{a}\ \underline{b}$

Applications
Think: Function Calls

$\underline{(\lambda x.\lambda y.\underline{x\ y})\ \underline{a\ b}}$

Example - Evaluation

$$(\lambda x. \quad \lambda y.x \quad y) \quad a \quad b$$

Example - Evaluation

$$(\lambda x. \qquad \lambda y.x \qquad y) \qquad a \qquad b \qquad \text{Substitute } x \mapsto a$$

Example - Evaluation

$(\lambda x. \quad \lambda y.x \quad y) \quad a \quad b$ Substitute $x \mapsto a$

Example - Evaluation

$$
\begin{array}{lcccccl}
 & (\lambda x. & \lambda y.x & y) & a & b & \text{Substitute } x \mapsto a \\
\rightarrow & & (\lambda y.a & y) & & b &
\end{array}
$$

Example - Evaluation

$$
\begin{array}{llllll}
 & (\lambda x. & \lambda y.x & y) & a & b & \text{Substitute } x \mapsto a \\
\rightarrow & & (\lambda y.a & y) & & b & \text{Substitute } y \mapsto b
\end{array}
$$

Example - Evaluation

$$\begin{array}{lllllll} & (\lambda x. & \lambda y.x & y) & a & b & \text{Substitute } x \mapsto a \\ \rightarrow & & (\lambda y.a & y) & & b & \text{Substitute } y \mapsto b \end{array}$$

Example - Evaluation

$$\begin{array}{lccccl} & (\lambda x. \; \lambda y.x & y) & a & b & \text{Substitute } x \mapsto a \\ \to & (\lambda y.a & y) & & b & \text{Substitute } y \mapsto b \\ \to & a & b & & & \end{array}$$

Notational Conventions

- We use parentheses to clearify what's meant
- Applications associate to the left

$$s\ t\ u \equiv (s\ t)\ u$$

- Abstractions expand as much to the right as possible

$$\lambda x.\lambda y.x\ y\ x \equiv \lambda x.(\lambda y.(x\ y\ x))$$

Scope

$$\lambda x.\lambda y.x\ y\ z$$

Bound* and *Free

λy y is *bound*, x and z are *free*

λx x and y are *bound*, z is *free*

λx, λy *binders*

Scope

$$\lambda x.\lambda y.x\ y\ z$$

Bound and _Free_

λy y is *bound*, x and z are *free*

λx x and y are *bound*, z is *free*

λx, λy *binders*

A term with no free variables is _closed_

- A *combinator*
- $id \equiv \lambda x.x$
- Y, S, K, I ...

Higher Order Functions

- Functions that take or return functions
  - Are there "by definition"

$$\underbrace{\underbrace{\lambda x.x}_{\textit{Abstraction}} \quad \underbrace{\lambda y.y}_{\textit{Abstraction}}}_{\textit{Application}} \rightarrow \underbrace{\lambda y.y}_{\textit{Abstraction}}$$

Currying

Idea

- Take a function with n arguments
- Create a function that takes one argument and returns a function with $n - 1$ arguments

Currying

Idea

- Take a function with n arguments
- Create a function that takes one argument and returns a function with $n-1$ arguments

Example

- `(+1)` Section in Haskell
- $(\lambda x.\lambda y. + x\ y)\ 1 \rightarrow \lambda y. +\ 1\ y$

Currying

Idea

- Take a function with n arguments
- Create a function that takes one argument and returns a function with $n - 1$ arguments

Example

- `(+1)` Section in Haskell
- $(\lambda x.\lambda y. + x\ y)\ 1 \rightarrow \lambda y. +\ 1\ y$

- Partial Function Application is there "by definition"
    - You can use this stunt to "curry" in every language that supports "Lambda Expressions"

Reductions and Conversions

Alpha Conversion

$$\lambda x.x \rightarrow_{\alpha} \lambda y.y$$

Reductions and Conversions

Alpha Conversion

$$\lambda x.x \rightarrow_\alpha \lambda y.y$$

Beta Reduction

$$(\lambda x.x)\ y \rightarrow_\beta y$$

Reductions and Conversions

Alpha Conversion

$$\lambda x.x \rightarrow_\alpha \lambda y.y$$

Beta Reduction

$$(\lambda x.x)\ y \rightarrow_\beta y$$

Eta Conversion
Iff (if and only if) x is not free in f:

$$\lambda x.f\ x \rightarrow_\eta f$$

$$(\lambda x.\underbrace{(\lambda y.y)}_{f}\ x)\ a \rightarrow_\eta \underbrace{(\lambda y.y)}_{f}\ a$$

Reductions and Conversions

Alpha Conversion

$$\lambda x.x \rightarrow_\alpha \lambda y.y$$

Beta Reduction

$$(\lambda x.x)\ y \rightarrow_\beta y$$

Eta Conversion

Iff (if and only if) x is not free in f:

$$\lambda x.f\ x \rightarrow_\eta f$$

$$(\lambda x.\underbrace{(\lambda y.y)}_{f}\ x)\ a \rightarrow_\eta \underbrace{(\lambda y.y)}_{f}\ a$$

If x is free in f, η conversion not possible:

$$\lambda x.\underbrace{(\lambda y.y\ \ \overset{\text{Bound}}{\overset{\downarrow}{x}}\ )}_{f}\ x \not\rightarrow_\eta (\lambda y.y\ \ \overset{\text{Free?!}}{\overset{\downarrow}{x}}\ )$$

Remarks

- Everything (Term) is an Expression
    - No statements
- No "destructive" Assignments
    - The reason why FP Languages promote pure functions
    - But you could invent a built-in function to manipulate "state"...

Evaluation

Operational Semantics

- We learned how to write down and talk about Lambda Calculus Terms
- How to evaluate them?
- Different Strategies
    - Interesting outcomes

Full Beta-Reduction

- RedEx
  - **Red**ucible **Ex**pression
  - Always an Application

$$\underbrace{(\lambda x.x)\ \underbrace{((\lambda y.y)\ (\lambda z.\underbrace{(\lambda a.a)\ z}_{\textit{RedEx}}))}_{\textit{RedEx}}}_{\textit{RedEx}}$$

Full Beta-Reduction

- RedEx
  - **Red**ucible **Ex**pression
  - Always an Application

$$\underbrace{(\lambda x.x)\ (\underbrace{(\lambda y.y)\ (\lambda z.\underbrace{(\lambda a.a)\ z}_{\textit{RedEx}})}_{\textit{RedEx}})}_{\textit{RedEx}}$$

Full Beta-Reduction

- Any RedEx, Any Time
- Like in Arithmetics
- Too vague for programming...

Normal Order Reduction

$$(\lambda x.x)\ ((\lambda y.y)\ (\lambda z.(\lambda a.a)\ z))$$

Normal Order Reduction

- Left-most, Outer-most RedEx

Normal Order Reduction

$$(\lambda x.x)\ ((\lambda y.y)\ (\lambda z.(\lambda a.a)\ z))$$

Normal Order Reduction

- Left-most, Outer-most RedEx

Normal Order Reduction

$$
\begin{aligned}
&(\lambda x.x)\ ((\lambda y.y)\ (\lambda z.(\lambda a.a)\ z)) \\
\rightarrow &(\lambda y.y)\ (\lambda z.(\lambda a.a)\ z)
\end{aligned}
$$

Normal Order Reduction

- Left-most, Outer-most RedEx

Normal Order Reduction

$$
\begin{aligned}
&(\lambda x.x)\ ((\lambda y.y)\ (\lambda z.(\lambda a.a)\ z)) \\
\rightarrow &(\lambda y.y)\ (\lambda z.(\lambda a.a)\ z)
\end{aligned}
$$

Normal Order Reduction

- Left-most, Outer-most RedEx

Normal Order Reduction

$$\begin{aligned}
&(\lambda x.x)\ ((\lambda y.y)\ (\lambda z.(\lambda a.a)\ z)) \\
\to&(\lambda y.y)\ (\lambda z.(\lambda a.a)\ z) \\
\to&\lambda z.(\lambda a.a)\ z
\end{aligned}$$

Normal Order Reduction

- Left-most, Outer-most RedEx

Normal Order Reduction

$$
\begin{aligned}
&(\lambda x.x)\ ((\lambda y.y)\ (\lambda z.(\lambda a.a)\ z)) \\
\to&(\lambda y.y)\ (\lambda z.(\lambda a.a)\ z) \\
\to&\lambda z.(\lambda a.a)\ z
\end{aligned}
$$

Normal Order Reduction

- Left-most, Outer-most RedEx

Normal Order Reduction

$$
\begin{aligned}
&(\lambda x.x)\ ((\lambda y.y)\ (\lambda z.(\lambda a.a)\ z)) \\
\rightarrow&(\lambda y.y)\ (\lambda z.(\lambda a.a)\ z) \\
\rightarrow&\lambda z.(\lambda a.a)\ z \\
\rightarrow&\lambda z.z
\end{aligned}
$$

Normal Order Reduction

- Left-most, Outer-most RedEx

Call-by-Name

$$(\lambda x.x)\ ((\lambda y.y)\ (\lambda z.(\lambda a.a)\ z))$$

Call-by-Name

- like Normal Order Reduction, but no reductions inside Abstractions
  - Abstractions are values
- lazy, non-strict
  - Parameters are not evaluated before they are used
- Optimization: Save results → *Call-by-Need*

$$(\lambda x.x)\ ((\lambda y.y)\ (\lambda z.(\lambda a.a)\ z))$$

Call-by-Name

- like Normal Order Reduction, but no reductions inside Abstractions
  - Abstractions are values
- lazy, non-strict
  - Parameters are not evaluated before they are used
- Optimization: Save results $\rightarrow$ *Call-by-Need*

Call-by-Name

$$(\lambda x.x)\ ((\lambda y.y)\ (\lambda z.(\lambda a.a)\ z))$$
$$\rightarrow (\lambda y.y)\ (\lambda z.(\lambda a.a)\ z)$$

Call-by-Name

- like Normal Order Reduction, but no reductions inside Abstractions
  - Abstractions are values
- lazy, non-strict
  - Parameters are not evaluated before they are used
- Optimization: Save results → *Call-by-Need*

Call-by-Name

$$(\lambda x.x)\ ((\lambda y.y)\ (\lambda z.(\lambda a.a)\ z))$$
$$\rightarrow (\lambda y.y)\ (\lambda z.(\lambda a.a)\ z)$$

Call-by-Name

- like Normal Order Reduction, but no reductions inside Abstractions
  - Abstractions are values
- lazy, non-strict
  - Parameters are not evaluated before they are used
- Optimization: Save results $\rightarrow$ *Call-by-Need*

Call-by-Name

$$
\begin{aligned}
&(\lambda x.x)\ ((\lambda y.y)\ (\lambda z.(\lambda a.a)\ z)) \\
\rightarrow &(\lambda y.y)\ (\lambda z.(\lambda a.a)\ z) \\
\rightarrow &\lambda z.(\lambda a.a)\ z
\end{aligned}
$$

Call-by-Name

- like Normal Order Reduction, but no reductions inside Abstractions
  - Abstractions are values
- lazy, non-strict
  - Parameters are not evaluated before they are used
- Optimization: Save results → *Call-by-Need*

Call-by-Name

$$
\begin{aligned}
& (\lambda x.x)\ ((\lambda y.y)\ (\lambda z.(\lambda a.a)\ z)) \\
\rightarrow & (\lambda y.y)\ (\lambda z.(\lambda a.a)\ z) \\
\rightarrow & \lambda z.(\lambda a.a)\ z \\
\not\rightarrow &
\end{aligned}
$$

Call-by-Name

- like Normal Order Reduction, but no reductions inside Abstractions
  - Abstractions are values
- lazy, non-strict
  - Parameters are not evaluated before they are used
- Optimization: Save results → *Call-by-Need*

Call-by-Value

$$(\lambda x.x)\ ((\lambda y.y)\ (\lambda z.(\lambda a.a)\ z))$$

Call-by-Value

- Outer-most, only if right-hand side was reduced to a value
- No reductions inside Abstractions
  - Abstractions are values
- eager, strict
  - Parameters are evaluated before they are used

Call-by-Value

$$(\lambda x.x)\ ((\lambda y.y)\ (\lambda z.(\lambda a.a)\ z))$$

Call-by-Value

- Outer-most, only if right-hand side was reduced to a value
- No reductions inside Abstractions
  - Abstractions are values
- eager, strict
  - Parameters are evaluated before they are used

Call-by-Value

$$
\begin{aligned}
&(\lambda x.x)\ ((\lambda y.y)\ (\lambda z.(\lambda a.a)\ z)) \\
\rightarrow &(\lambda x.x)\ (\lambda z.(\lambda a.a)\ z)
\end{aligned}
$$

Call-by-Value

- Outer-most, only if right-hand side was reduced to a value
- No reductions inside Abstractions
  - Abstractions are values
- eager, strict
  - Parameters are evaluated before they are used

Call-by-Value

$$(\lambda x.x)\ ((\lambda y.y)\ (\lambda z.(\lambda a.a)\ z))$$
$$\rightarrow (\lambda x.x)\ (\lambda z.(\lambda a.a)\ z)$$

Call-by-Value

- Outer-most, only if right-hand side was reduced to a value
- No reductions inside Abstractions
  - Abstractions are values
- eager, strict
  - Parameters are evaluated before they are used

Call-by-Value

$$
\begin{aligned}
&(\lambda x.x)\ ((\lambda y.y)\ (\lambda z.(\lambda a.a)\ z)) \\
\rightarrow&(\lambda x.x)\ (\lambda z.(\lambda a.a)\ z) \\
\rightarrow&\lambda z.(\lambda a.a)\ z
\end{aligned}
$$

Call-by-Value

- Outer-most, only if right-hand side was reduced to a value
- No reductions inside Abstractions
    - Abstractions are values
- eager, strict
    - Parameters are evaluated before they are used

Call-by-Value

$$
\begin{aligned}
& (\lambda x.x)\ ((\lambda y.y)\ (\lambda z.(\lambda a.a)\ z)) \\
\to & (\lambda x.x)\ (\lambda z.(\lambda a.a)\ z) \\
\to & \lambda z.(\lambda a.a)\ z \\
\not\to &
\end{aligned}
$$

Call-by-Value

- Outer-most, only if right-hand side was reduced to a value
- No reductions inside Abstractions
    - Abstractions are values
- eager, strict
    - Parameters are evaluated before they are used

Church Encodings

Church Encodings

- Encode Data into the Lambda Calculus
- To simplify our formulas, let's say that we have declarations

$$id \equiv \lambda x.x$$

$$id\ y \rightarrow y$$

Booleans

$$\begin{aligned} \mathit{true} &\equiv \lambda t.\lambda f.t \\ \mathit{false} &\equiv \lambda t.\lambda f.f \end{aligned}$$

Booleans

$$
\begin{aligned}
\mathit{true} &\equiv \lambda t.\lambda f.t \\
\mathit{false} &\equiv \lambda t.\lambda f.f
\end{aligned}
$$

$$
\mathit{test} \equiv \lambda c.\lambda t.\lambda f.c\ t\ f
$$

Booleans

$$test\ \ true\ \ a\ \ b$$

$$
\begin{aligned}
true &\equiv \lambda t.\lambda f.t \\
false &\equiv \lambda t.\lambda f.f
\end{aligned}
$$

$$test \equiv \lambda c.\lambda t.\lambda f.c\ \ t\ \ f$$

Booleans

$$test\ true\ a\ b$$

$$\begin{aligned} true &\equiv \lambda t.\lambda f.t \\ false &\equiv \lambda t.\lambda f.f \end{aligned}$$

$$test \equiv \lambda c.\lambda t.\lambda f.c\ t\ f$$

Booleans

$$\begin{aligned} true &\equiv \lambda t.\lambda f.t \\ false &\equiv \lambda t.\lambda f.f \\ \\ test &\equiv \lambda c.\lambda t.\lambda f.c\ t\ f \end{aligned}$$

$$\begin{aligned} &test\ true\ a\ b \\ \equiv\ &(\lambda c.\lambda t.\lambda f.c\ t\ f)\ true\ a\ b \end{aligned}$$

Booleans

$$true \equiv \lambda t.\lambda f.t$$

$$false \equiv \lambda t.\lambda f.f$$

$$test \equiv \lambda c.\lambda t.\lambda f.c\ t\ f$$

$$\begin{aligned} & test\ true\ a\ b \\ \equiv\ & (\lambda c.\lambda t.\lambda f.c\ t\ f)\ true\ a\ b \end{aligned}$$

Booleans

$$\begin{aligned} \mathit{true} &\equiv \lambda t.\lambda f.t \\ \mathit{false} &\equiv \lambda t.\lambda f.f \end{aligned}$$

$$\mathit{test} \equiv \lambda c.\lambda t.\lambda f.c\ t\ f$$

$$\begin{aligned} &\mathit{test}\ \mathit{true}\ a\ b \\ \equiv\ &(\lambda c.\lambda t.\lambda f.c\ t\ f)\ \mathit{true}\ a\ b \\ \rightarrow\ &(\lambda t.\lambda f.\mathit{true}\ t\ f)\ a\ b \end{aligned}$$

Booleans

$$\textit{true} \equiv \lambda t.\lambda f.t$$

$$\textit{false} \equiv \lambda t.\lambda f.f$$

$$\textit{test} \equiv \lambda c.\lambda t.\lambda f.c\ t\ f$$

$$\begin{aligned} &\textit{test}\ \textit{true}\ a\ b \\ \equiv\ &(\lambda c.\lambda t.\lambda f.c\ t\ f)\ \textit{true}\ a\ b \\ \rightarrow\ &(\lambda t.\lambda f.\textit{true}\ t\ f)\ a\ b \end{aligned}$$

Booleans

$$\begin{aligned} true &\equiv \lambda t.\lambda f.t \\ false &\equiv \lambda t.\lambda f.f \\ \\ test &\equiv \lambda c.\lambda t.\lambda f.c\ t\ f \end{aligned}$$

$$\begin{aligned} & test\ true\ a\ b \\ \equiv\ & (\lambda c.\lambda t.\lambda f.c\ t\ f)\ true\ a\ b \\ \to\ & (\lambda t.\lambda f.true\ t\ f)\ a\ b \\ \to\ & (\lambda f.true\ a\ f)\ b \end{aligned}$$

Booleans

$$\begin{aligned} true &\equiv \lambda t.\lambda f.t \\ false &\equiv \lambda t.\lambda f.f \end{aligned}$$

$$test \equiv \lambda c.\lambda t.\lambda f.c\ t\ f$$

$$\begin{aligned} & test\ true\ a\ b \\ \equiv\ & (\lambda c.\lambda t.\lambda f.c\ t\ f)\ true\ a\ b \\ \rightarrow\ & (\lambda t.\lambda f.true\ t\ f)\ a\ b \\ \rightarrow\ & (\lambda f.true\ a\ f)\ b \end{aligned}$$

Booleans

$$\begin{aligned} true &\equiv \lambda t.\lambda f.t \\ false &\equiv \lambda t.\lambda f.f \\ \\ test &\equiv \lambda c.\lambda t.\lambda f.c\ t\ f \end{aligned}$$

$$\begin{aligned} & test\ true\ a\ b \\ \equiv\ & (\lambda c.\lambda t.\lambda f.c\ t\ f)\ true\ a\ b \\ \rightarrow\ & (\lambda t.\lambda f.true\ t\ f)\ a\ b \\ \rightarrow\ & (\lambda f.true\ a\ f)\ b \\ \rightarrow\ & true\ a\ b \end{aligned}$$

Booleans

$$\begin{aligned} true &\equiv \lambda t.\lambda f.t \\ false &\equiv \lambda t.\lambda f.f \\ \\ test &\equiv \lambda c.\lambda t.\lambda f.c\ t\ f \end{aligned}$$

$$\begin{aligned} & test\ true\ a\ b \\ \equiv\ & (\lambda c.\lambda t.\lambda f.c\ t\ f)\ true\ a\ b \\ \rightarrow\ & (\lambda t.\lambda f.true\ t\ f)\ a\ b \\ \rightarrow\ & (\lambda f.true\ a\ f)\ b \\ \rightarrow\ & true\ a\ b \end{aligned}$$

Booleans

$$\begin{aligned} true &\equiv \lambda t.\lambda f.t \\ false &\equiv \lambda t.\lambda f.f \\ \\ test &\equiv \lambda c.\lambda t.\lambda f.c\ t\ f \end{aligned}$$

$$\begin{aligned} & test\ true\ a\ b \\ \equiv\ & (\lambda c.\lambda t.\lambda f.c\ t\ f)\ true\ a\ b \\ \rightarrow\ & (\lambda t.\lambda f.true\ t\ f)\ a\ b \\ \rightarrow\ & (\lambda f.true\ a\ f)\ b \\ \rightarrow\ & true\ a\ b \\ \equiv\ & (\lambda t.\lambda f.t)\ a\ b \end{aligned}$$

Booleans

$$
\begin{aligned}
\mathit{true} &\equiv \lambda t.\lambda f.t \\
\mathit{false} &\equiv \lambda t.\lambda f.f \\
\\
\mathit{test} &\equiv \lambda c.\lambda t.\lambda f.c\ t\ f
\end{aligned}
$$

$$
\begin{aligned}
&\mathit{test}\ \mathit{true}\ a\ b \\
\equiv\ &(\lambda c.\lambda t.\lambda f.c\ t\ f)\ \mathit{true}\ a\ b \\
\rightarrow\ &(\lambda t.\lambda f.\mathit{true}\ t\ f)\ a\ b \\
\rightarrow\ &(\lambda f.\mathit{true}\ a\ f)\ b \\
\rightarrow\ &\mathit{true}\ a\ b \\
\equiv\ &(\lambda t.\lambda f.t)\ a\ b
\end{aligned}
$$

Booleans

$$\begin{aligned} true &\equiv \lambda t.\lambda f.t \\ false &\equiv \lambda t.\lambda f.f \\ \\ test &\equiv \lambda c.\lambda t.\lambda f.c\ t\ f \end{aligned}$$

$$\begin{aligned} & test\ true\ a\ b \\ \equiv\ & (\lambda c.\lambda t.\lambda f.c\ t\ f)\ true\ a\ b \\ \rightarrow\ & (\lambda t.\lambda f.true\ t\ f)\ a\ b \\ \rightarrow\ & (\lambda f.true\ a\ f)\ b \\ \rightarrow\ & true\ a\ b \\ \equiv\ & (\lambda t.\lambda f.t)\ a\ b \\ \rightarrow\ & (\lambda f.a)\ b \end{aligned}$$

Booleans

$$true \equiv \lambda t.\lambda f.t$$

$$false \equiv \lambda t.\lambda f.f$$

$$test \equiv \lambda c.\lambda t.\lambda f.c\ t\ f$$

$$\begin{aligned}
& test\ true\ a\ b \\
\equiv\ & (\lambda c.\lambda t.\lambda f.c\ t\ f)\ true\ a\ b \\
\rightarrow\ & (\lambda t.\lambda f.true\ t\ f)\ a\ b \\
\rightarrow\ & (\lambda f.true\ a\ f)\ b \\
\rightarrow\ & true\ a\ b \\
\equiv\ & (\lambda t.\lambda f.t)\ a\ b \\
\rightarrow\ & (\lambda f.a)\ b
\end{aligned}$$

Booleans

$$
\begin{aligned}
\mathit{true} &\equiv \lambda t.\lambda f.t \\
\mathit{false} &\equiv \lambda t.\lambda f.f \\
\\
\mathit{test} &\equiv \lambda c.\lambda t.\lambda f.c\ t\ f
\end{aligned}
$$

$$
\begin{aligned}
& \mathit{test}\ \mathit{true}\ a\ b \\
\equiv\ & (\lambda c.\lambda t.\lambda f.c\ t\ f)\ \mathit{true}\ a\ b \\
\rightarrow\ & (\lambda t.\lambda f.\mathit{true}\ t\ f)\ a\ b \\
\rightarrow\ & (\lambda f.\mathit{true}\ a\ f)\ b \\
\rightarrow\ & \mathit{true}\ a\ b \\
\equiv\ & (\lambda t.\lambda f.t)\ a\ b \\
\rightarrow\ & (\lambda f.a)\ b \\
\rightarrow\ & a
\end{aligned}
$$

And

$$\mathit{true} \equiv \lambda t.\lambda f.t$$

$$\mathit{false} \equiv \lambda t.\lambda f.f$$

And

$$\begin{aligned} true &\equiv \lambda t.\lambda f.t \\ false &\equiv \lambda t.\lambda f.f \end{aligned}$$

$$and \equiv \lambda p.\lambda q.p\ q\ p$$

And

$$and\ \ true\ \ false$$

$$
\begin{aligned}
true &\equiv \qquad \lambda t.\lambda f.t \\
false &\equiv \qquad \lambda t.\lambda f.f
\end{aligned}
$$

$$and \equiv \quad \lambda p.\lambda q.p\ q\ p$$

And

$$\textit{and} \;\; \textit{true} \;\; \textit{false}$$

$$\begin{aligned} \textit{true} &\equiv \lambda t.\lambda f.t \\ \textit{false} &\equiv \lambda t.\lambda f.f \end{aligned}$$

$$\textit{and} \equiv \lambda p.\lambda q.p\ q\ p$$

And

$$\mathit{true} \equiv \lambda t.\lambda f.t$$

$$\mathit{false} \equiv \lambda t.\lambda f.f$$

$$\mathit{and} \equiv \lambda p.\lambda q.p\ q\ p$$

$$\begin{aligned} &\mathit{and}\ \mathit{true}\ \mathit{false} \\ \equiv&(\lambda p.\lambda q.p\ q\ p)\ \mathit{true}\ \mathit{false} \end{aligned}$$

And

$$\mathit{true} \equiv \lambda t.\lambda f.t$$

$$\mathit{false} \equiv \lambda t.\lambda f.f$$

$$\mathit{and} \equiv \lambda p.\lambda q.p\ q\ p$$

$$\begin{aligned} &\mathit{and}\ \mathit{true}\ \mathit{false} \\ \equiv&(\lambda p.\lambda q.p\ q\ p)\ \mathit{true}\ \mathit{false} \end{aligned}$$

And

$$\begin{aligned} \mathit{true} &\equiv \lambda t.\lambda f.t \\ \mathit{false} &\equiv \lambda t.\lambda f.f \end{aligned}$$

$$\mathit{and} \equiv \lambda p.\lambda q.p\ q\ p$$

$$\begin{aligned} &\mathit{and}\ \mathit{true}\ \mathit{false} \\ \equiv&(\lambda p.\lambda q.p\ q\ p)\ \mathit{true}\ \mathit{false} \\ \rightarrow&(\lambda q.\mathit{true}\ q\ \mathit{true})\ \mathit{false} \end{aligned}$$

And

$$
\begin{aligned}
true &\equiv \lambda t.\lambda f.t \\
false &\equiv \lambda t.\lambda f.f \\
\\
and &\equiv \lambda p.\lambda q.p\ q\ p
\end{aligned}
$$

$$
\begin{aligned}
& and\ true\ false \\
\equiv & (\lambda p.\lambda q.p\ q\ p)\ true\ false \\
\rightarrow & (\lambda q.true\ q\ true)\ false
\end{aligned}
$$

And

$$
\begin{aligned}
\mathit{true} &\equiv \lambda t.\lambda f.t \\
\mathit{false} &\equiv \lambda t.\lambda f.f
\end{aligned}
$$

$$
\mathit{and} \equiv \lambda p.\lambda q.p\ q\ p
$$

$$
\begin{aligned}
&\mathit{and}\ \mathit{true}\ \mathit{false} \\
\equiv&(\lambda p.\lambda q.p\ q\ p)\ \mathit{true}\ \mathit{false} \\
\rightarrow&(\lambda q.\mathit{true}\ q\ \mathit{true})\ \mathit{false} \\
\rightarrow&\mathit{true}\ \mathit{false}\ \mathit{true}
\end{aligned}
$$

And

$$\begin{aligned} true &\equiv \lambda t.\lambda f.t \\ false &\equiv \lambda t.\lambda f.f \\ \\ and &\equiv \lambda p.\lambda q.p\ q\ p \end{aligned}$$

$$\begin{aligned} & and\ true\ false \\ \equiv & (\lambda p.\lambda q.p\ q\ p)\ true\ false \\ \to & (\lambda q.true\ q\ true)\ false \\ \to & true\ false\ true \end{aligned}$$

And

$$\begin{aligned} true &\equiv \lambda t.\lambda f.t \\ false &\equiv \lambda t.\lambda f.f \\ \\ and &\equiv \lambda p.\lambda q.p\ q\ p \end{aligned}$$

$$\begin{aligned} & and\ true\ false \\ \equiv & (\lambda p.\lambda q.p\ q\ p)\ true\ false \\ \rightarrow & (\lambda q.true\ q\ true)\ false \\ \rightarrow & true\ false\ true \\ \equiv & (\lambda t.\lambda f.t)\ false\ true \end{aligned}$$

And

$$\begin{aligned} \mathit{true} &\equiv \lambda t.\lambda f.t \\ \mathit{false} &\equiv \lambda t.\lambda f.f \\ \\ \mathit{and} &\equiv \lambda p.\lambda q.p\ q\ p \end{aligned}$$

$$\begin{aligned} & \mathit{and}\ \mathit{true}\ \mathit{false} \\ \equiv & (\lambda p.\lambda q.p\ q\ p)\ \mathit{true}\ \mathit{false} \\ \rightarrow & (\lambda q.\mathit{true}\ q\ \mathit{true})\ \mathit{false} \\ \rightarrow & \mathit{true}\ \mathit{false}\ \mathit{true} \\ \equiv & (\lambda t.\lambda f.t)\ \mathit{false}\ \mathit{true} \end{aligned}$$

And

$$\begin{aligned} \mathit{true} &\equiv \lambda t.\lambda f.t \\ \mathit{false} &\equiv \lambda t.\lambda f.f \\ \\ \mathit{and} &\equiv \lambda p.\lambda q.p\ q\ p \end{aligned}$$

$$\begin{aligned} & \mathit{and}\ \mathit{true}\ \mathit{false} \\ \equiv & (\lambda p.\lambda q.p\ q\ p)\ \mathit{true}\ \mathit{false} \\ \rightarrow & (\lambda q.\mathit{true}\ q\ \mathit{true})\ \mathit{false} \\ \rightarrow & \mathit{true}\ \mathit{false}\ \mathit{true} \\ \equiv & (\lambda t.\lambda f.t)\ \mathit{false}\ \mathit{true} \\ \rightarrow & (\lambda f.\mathit{false})\ \mathit{true} \end{aligned}$$

And

$$\begin{aligned} true &\equiv \lambda t.\lambda f.t \\ false &\equiv \lambda t.\lambda f.f \\ \\ and &\equiv \lambda p.\lambda q.p\ q\ p \end{aligned}$$

$$\begin{aligned} & and\ true\ false \\ \equiv & (\lambda p.\lambda q.p\ q\ p)\ true\ false \\ \rightarrow & (\lambda q.true\ q\ true)\ false \\ \rightarrow & true\ false\ true \\ \equiv & (\lambda t.\lambda f.t)\ false\ true \\ \rightarrow & (\lambda f.false)\ true \end{aligned}$$

And

$$\begin{aligned} \mathit{true} &\equiv \lambda t.\lambda f.t \\ \mathit{false} &\equiv \lambda t.\lambda f.f \\[1em] \mathit{and} &\equiv \lambda p.\lambda q.p\ q\ p \end{aligned}$$

$$\begin{aligned} & \mathit{and}\ \mathit{true}\ \mathit{false} \\ \equiv & (\lambda p.\lambda q.p\ q\ p)\ \mathit{true}\ \mathit{false} \\ \rightarrow & (\lambda q.\mathit{true}\ q\ \mathit{true})\ \mathit{false} \\ \rightarrow & \mathit{true}\ \mathit{false}\ \mathit{true} \\ \equiv & (\lambda t.\lambda f.t)\ \mathit{false}\ \mathit{true} \\ \rightarrow & (\lambda f.\mathit{false})\ \mathit{true} \\ \rightarrow & \mathit{false} \end{aligned}$$

Or

$$\lambda p.\lambda q.p\ p\ q$$

Pairs

$$\mathit{pair} \equiv \quad \lambda x.\lambda y.\lambda z.z\ x\ y$$

Pairs

$$\begin{aligned} \mathit{pair} &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\ \mathit{first} &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\ \mathit{second} &\equiv (\lambda p.p)\ \lambda x.\lambda y.y \end{aligned}$$

Pairs

$$pair_{AB} \equiv \qquad\qquad pair\ \ a\ \ b$$

$$\begin{aligned} pair &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\ first &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\ second &\equiv (\lambda p.p)\ \lambda x.\lambda y.y \end{aligned}$$

Pairs

$$pair_{AB} \equiv \qquad\qquad \textit{pair}\ a\ b$$

$$\begin{aligned} pair &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\ first &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\ second &\equiv (\lambda p.p)\ \lambda x.\lambda y.y \end{aligned}$$

Pairs

$$\begin{aligned} pair &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\ first &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\ second &\equiv (\lambda p.p)\ \lambda x.\lambda y.y \end{aligned}$$

$$\begin{aligned} pair_{AB} &\equiv \textcolor{orange}{pair}\ a\ b \\ &\equiv (\lambda x.\lambda y.\lambda z.z\ x\ y)\ a\ b \end{aligned}$$

Pairs

$$\begin{aligned} pair_{AB} &\equiv \quad pair\ a\ b \\ &\equiv \quad (\lambda x.\lambda y.\lambda z.z\ x\ y)\ a\ b \end{aligned}$$

$$\begin{aligned} pair &\equiv \quad \lambda x.\lambda y.\lambda z.z\ x\ y \\ first &\equiv \quad (\lambda p.p)\ \lambda x.\lambda y.x \\ second &\equiv \quad (\lambda p.p)\ \lambda x.\lambda y.y \end{aligned}$$

Pairs

$$\begin{aligned} pair &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\ first &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\ second &\equiv (\lambda p.p)\ \lambda x.\lambda y.y \end{aligned}$$

$$\begin{aligned} pair_{AB} &\equiv pair\ a\ b \\ &\equiv (\lambda x.\lambda y.\lambda z.z\ x\ y)\ a\ b \\ &\rightarrow (\lambda y.\lambda z.z\ a\ y)\ b \end{aligned}$$

Pairs

$$\begin{aligned} pair &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\ first &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\ second &\equiv (\lambda p.p)\ \lambda x.\lambda y.y \end{aligned}$$

$$\begin{aligned} pair_{AB} &\equiv pair\ a\ b \\ &\equiv (\lambda x.\lambda y.\lambda z.z\ x\ y)\ a\ b \\ &\rightarrow (\lambda y.\lambda z.z\ a\ y)\ b \end{aligned}$$

Pairs

$$\begin{aligned} pair &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\ first &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\ second &\equiv (\lambda p.p)\ \lambda x.\lambda y.y \end{aligned}$$

$$\begin{aligned} pair_{AB} &\equiv & pair\ a\ b \\ &\equiv & (\lambda x.\lambda y.\lambda z.z\ x\ y)\ a\ b \\ &\rightarrow & (\lambda y.\lambda z.z\ a\ y)\ b \\ &\rightarrow & \lambda z.z\ a\ b \end{aligned}$$

Pairs

$$
\begin{aligned}
pair &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\
first &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\
second &\equiv (\lambda p.p)\ \lambda x.\lambda y.y
\end{aligned}
$$

$$
\begin{aligned}
pair_{AB} &\equiv pair\ a\ b \\
&\equiv (\lambda x.\lambda y.\lambda z.z\ x\ y)\ a\ b \\
&\rightarrow (\lambda y.\lambda z.z\ a\ y)\ b \\
&\rightarrow \lambda z.z\ a\ b \\
&\equiv pair'_{ab}
\end{aligned}
$$

Pairs (continued)

$$\begin{aligned}
\mathit{pair} &\equiv \quad \lambda x.\lambda y.\lambda z.z\ x\ y \\
\mathit{first} &\equiv \quad (\lambda p.p)\ \lambda x.\lambda y.x \\
\mathit{pair}'_{ab} &\equiv \quad \lambda z.z\ a\ b
\end{aligned}$$

Pairs (continued)

$$\mathit{first}\ \mathit{pair}'_{ab}$$

$$\begin{aligned} \mathit{pair} &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\ \mathit{first} &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\ \mathit{pair}'_{ab} &\equiv \lambda z.z\ a\ b \end{aligned}$$

Pairs (continued)

$$\textit{first}\ \textit{pair}'_{ab}$$

$$\begin{aligned}
\textit{pair} &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\
\textit{first} &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\
\textit{pair}'_{ab} &\equiv \lambda z.z\ a\ b
\end{aligned}$$

Pairs (continued)

$$\begin{aligned} pair &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\ first &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\ pair'_{ab} &\equiv \lambda z.z\ a\ b \end{aligned}$$

$$\begin{aligned} & first\ pair'_{ab} \\ \equiv\ & (\lambda p.p)\ (\lambda x.\lambda y.x)\ pair'_{ab} \end{aligned}$$

Pairs (continued)

$$\begin{aligned} pair &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\ first &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\ pair'_{ab} &\equiv \lambda z.z\ a\ b \end{aligned}$$

$$\begin{aligned} & first\ pair'_{ab} \\ \equiv\ & (\lambda p.p)\ (\lambda x.\lambda y.x)\ pair'_{ab} \end{aligned}$$

Pairs (continued)

$$\begin{aligned} pair &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\ first &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\ pair'_{ab} &\equiv \lambda z.z\ a\ b \end{aligned}$$

$$\begin{aligned} & first\ pair'_{ab} \\ \equiv\ & (\lambda p.p)\ (\lambda x.\lambda y.x)\ pair'_{ab} \\ \rightarrow\ & pair'_{ab}\ (\lambda x.\lambda y.x) \end{aligned}$$

Pairs (continued)

$$\begin{aligned} pair &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\ first &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\ pair'_{ab} &\equiv \lambda z.z\ a\ b \end{aligned}$$

$$\begin{aligned} & first\ pair'_{ab} \\ \equiv\ & (\lambda p.p)\ (\lambda x.\lambda y.x)\ pair'_{ab} \\ \rightarrow\ & pair'_{ab}\ (\lambda x.\lambda y.x) \end{aligned}$$

Pairs (continued)

$$\begin{aligned} pair &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\ first &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\ pair'_{ab} &\equiv \lambda z.z\ a\ b \end{aligned}$$

$$\begin{aligned} & first\ pair'_{ab} \\ \equiv\quad & (\lambda p.p)\ (\lambda x.\lambda y.x)\ pair'_{ab} \\ \rightarrow\quad & pair'_{ab}\ (\lambda x.\lambda y.x) \\ \equiv\quad & (\lambda z.z\ a\ b)\ (\lambda x.\lambda y.x) \end{aligned}$$

Pairs (continued)

$$\begin{aligned} pair &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\ first &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\ pair'_{ab} &\equiv \lambda z.z\ a\ b \end{aligned}$$

$$\begin{aligned} & first\ pair'_{ab} \\ \equiv\ & (\lambda p.p)\ (\lambda x.\lambda y.x)\ pair'_{ab} \\ \rightarrow\ & pair'_{ab}\ (\lambda x.\lambda y.x) \\ \equiv\ & (\lambda z.z\ a\ b)\ (\lambda x.\lambda y.x) \end{aligned}$$

Pairs (continued)

$$\begin{aligned} pair &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\ first &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\ pair'_{ab} &\equiv \lambda z.z\ a\ b \end{aligned}$$

$$\begin{aligned} & first\ pair'_{ab} \\ \equiv\ & (\lambda p.p)\ (\lambda x.\lambda y.x)\ pair'_{ab} \\ \to\ & pair'_{ab}\ (\lambda x.\lambda y.x) \\ \equiv\ & (\lambda z.z\ a\ b)\ (\lambda x.\lambda y.x) \\ \to\ & (\lambda x.\lambda y.x)\ a\ b \end{aligned}$$

Pairs (continued)

$$\begin{aligned} pair &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\ first &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\ pair'_{ab} &\equiv \lambda z.z\ a\ b \end{aligned}$$

$$\begin{aligned} & first\ pair'_{ab} \\ \equiv\ & (\lambda p.p)\ (\lambda x.\lambda y.x)\ pair'_{ab} \\ \rightarrow\ & pair'_{ab}\ (\lambda x.\lambda y.x) \\ \equiv\ & (\lambda z.z\ a\ b)\ (\lambda x.\lambda y.x) \\ \rightarrow\ & (\lambda x.\lambda y.x)\ a\ b \end{aligned}$$

Pairs (continued)

$$
\begin{aligned}
pair &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\
first &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\
pair'_{ab} &\equiv \lambda z.z\ a\ b
\end{aligned}
$$

$$
\begin{aligned}
& first\ pair'_{ab} \\
\equiv\ & (\lambda p.p)\ (\lambda x.\lambda y.x)\ pair'_{ab} \\
\to\ & pair'_{ab}\ (\lambda x.\lambda y.x) \\
\equiv\ & (\lambda z.z\ a\ b)\ (\lambda x.\lambda y.x) \\
\to\ & (\lambda x.\lambda y.x)\ a\ b \\
\to\ & (\lambda y.a)\ b
\end{aligned}
$$

Pairs (continued)

$$
\begin{aligned}
pair &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\
first &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\
pair'_{ab} &\equiv \lambda z.z\ a\ b
\end{aligned}
$$

$$
\begin{aligned}
& first\ pair'_{ab} \\
\equiv\ & (\lambda p.p)\ (\lambda x.\lambda y.x)\ pair'_{ab} \\
\rightarrow\ & pair'_{ab}\ (\lambda x.\lambda y.x) \\
\equiv\ & (\lambda z.z\ a\ b)\ (\lambda x.\lambda y.x) \\
\rightarrow\ & (\lambda x.\lambda y.x)\ a\ b \\
\rightarrow\ & (\lambda y.a)\ b
\end{aligned}
$$

Pairs (continued)

$$
\begin{aligned}
pair &\equiv \lambda x.\lambda y.\lambda z.z\ x\ y \\
first &\equiv (\lambda p.p)\ \lambda x.\lambda y.x \\
pair'_{ab} &\equiv \lambda z.z\ a\ b
\end{aligned}
$$

$$
\begin{aligned}
& first\ pair'_{ab} \\
\equiv\ & (\lambda p.p)\ (\lambda x.\lambda y.x)\ pair'_{ab} \\
\rightarrow\ & pair'_{ab}\ (\lambda x.\lambda y.x) \\
\equiv\ & (\lambda z.z\ a\ b)\ (\lambda x.\lambda y.x) \\
\rightarrow\ & (\lambda x.\lambda y.x)\ a\ b \\
\rightarrow\ & (\lambda y.a)\ b \\
\rightarrow\ & a
\end{aligned}
$$

Peano Axioms
Every natural number can be defined with 0 and a successor function

$$
\begin{aligned}
0 &\equiv \lambda f.\lambda x.x \\
1 &\equiv \lambda f.\lambda x.f\ x \\
2 &\equiv \lambda f.\lambda x.f\ (f\ x) \\
3 &\equiv \lambda f.\lambda x.f\ (f\ (f\ x))
\end{aligned}
$$

Meaning

0 f is evaluated 0 times

1 f is evaluated once

x can be every lambda term

Numerals Example - Successor

$$0 \equiv \lambda f.\lambda x.x$$

$$1 \equiv \lambda f.\lambda x.f\ x$$

Numerals Example - Successor

$$
\begin{aligned}
0 &\equiv \lambda f.\lambda x.x \\
1 &\equiv \lambda f.\lambda x.f\ x
\end{aligned}
$$

$$
\mathit{successor} \equiv \lambda n.\lambda f.\lambda x.f\ (n\ f\ x)
$$

Numerals Example - Successor

successor 1

$$
\begin{aligned}
0 &\equiv \lambda f.\lambda x.x \\
1 &\equiv \lambda f.\lambda x.f\ x
\end{aligned}
$$

$$
\textit{successor} \equiv \lambda n.\lambda f.\lambda x.f\ (n\ f\ x)
$$

Numerals Example - Successor

successor 1

$$0 \equiv \lambda f.\lambda x.x$$
$$1 \equiv \lambda f.\lambda x.f\ x$$

$$\mathit{successor} \equiv \lambda n.\lambda f.\lambda x.f\ (n\ f\ x)$$

Numerals Example - Successor

$$
\begin{aligned}
0 &\equiv \lambda f.\lambda x.x \\
1 &\equiv \lambda f.\lambda x.f\ x
\end{aligned}
$$

$$
\textit{successor} \equiv \lambda n.\lambda f.\lambda x.f\ (n\ f\ x)
$$

$$
\begin{aligned}
&\textit{successor}\ 1 \\
\equiv\ &(\lambda n.\lambda f.\lambda x.f\ (n\ f\ x))\ 1
\end{aligned}
$$

Numerals Example - Successor

$$\begin{aligned} 0 &\equiv \lambda f.\lambda x.x \\ 1 &\equiv \lambda f.\lambda x.f\ x \end{aligned}$$

$$\begin{aligned} & \textit{successor}\ 1 \\ \equiv\ & (\lambda n.\lambda f.\lambda x.f\ (n\ f\ x))\ 1 \end{aligned}$$

$$\textit{successor} \equiv \lambda n.\lambda f.\lambda x.f\ (n\ f\ x)$$

Numerals Example - Successor

$$0 \equiv \lambda f.\lambda x.x$$

$$1 \equiv \lambda f.\lambda x.f\ x$$

$$
\begin{aligned}
& \mathit{successor}\ 1 \\
\equiv\ & (\lambda n.\lambda f.\lambda x.f\ (n\ f\ x))\ 1 \\
\rightarrow\ & \lambda f.\lambda x.f\ (1\ f\ x)
\end{aligned}
$$

$$\mathit{successor} \equiv \lambda n.\lambda f.\lambda x.f\ (n\ f\ x)$$

Numerals Example - Successor

$$
\begin{aligned}
0 &\equiv \lambda f.\lambda x.x \\
1 &\equiv \lambda f.\lambda x.f\ x
\end{aligned}
\qquad
\begin{aligned}
& & \textit{successor}\ 1 \\
&\equiv & (\lambda n.\lambda f.\lambda x.f\ (n\ f\ x))\ 1 \\
&\rightarrow & \lambda f.\lambda x.f\ (1\ f\ x)
\end{aligned}
$$

$$\textit{successor} \equiv \lambda n.\lambda f.\lambda x.f\ (n\ f\ x)$$

Note
We use *Normal Order Reduction* to reduce inside abstractions!

Numerals Example - Successor

$$\begin{aligned} 0 &\equiv \lambda f.\lambda x.x \\ 1 &\equiv \lambda f.\lambda x.f\ x \end{aligned}$$

$$\begin{aligned} & \mathit{successor}\ 1 \\ \equiv\ & (\lambda n.\lambda f.\lambda x.f\ (n\ f\ x))\ 1 \\ \rightarrow\ & \lambda f.\lambda x.f\ (1\ f\ x) \\ \equiv\ & \lambda f.\lambda x.f\ ((\lambda f.\lambda x.f\ x)\ f\ x) \end{aligned}$$

$$\mathit{successor} \equiv \lambda n.\lambda f.\lambda x.f\ (n\ f\ x)$$

Note
We use *Normal Order Reduction* to reduce inside abstractions!

Numerals Example - Successor

$$
\begin{aligned}
0 &\equiv \lambda f.\lambda x.x \\
1 &\equiv \lambda f.\lambda x.f\ x
\end{aligned}
$$

$$
\begin{aligned}
& \mathit{successor}\ 1 \\
\equiv\ & (\lambda n.\lambda f.\lambda x.f\ (n\ f\ x))\ 1 \\
\rightarrow\ & \lambda f.\lambda x.f\ (1\ f\ x) \\
\equiv\ & \lambda f.\lambda x.f\ ((\lambda f.\lambda x.f\ x)\ f\ x)
\end{aligned}
$$

$$
\mathit{successor} \equiv \lambda n.\lambda f.\lambda x.f\ (n\ f\ x)
$$

Note
We use *Normal Order Reduction* to reduce inside abstractions!

Numerals Example - Successor

$$
\begin{aligned}
0 &\equiv \lambda f.\lambda x.x \\
1 &\equiv \lambda f.\lambda x.f\ x \\
\\
\mathit{successor} &\equiv \lambda n.\lambda f.\lambda x.f\ (n\ f\ x)
\end{aligned}
$$

$$
\begin{aligned}
& \mathit{successor}\ 1 \\
\equiv\ & (\lambda n.\lambda f.\lambda x.f\ (n\ f\ x))\ 1 \\
\rightarrow\ & \lambda f.\lambda x.f\ (1\ f\ x) \\
\equiv\ & \lambda f.\lambda x.f\ ((\lambda f.\lambda x.f\ x)\ f\ x) \\
\rightarrow\ & \lambda f.\lambda x.f\ ((\lambda x.f\ x)\ x)
\end{aligned}
$$

Note
We use *Normal Order Reduction* to reduce inside abstractions!

Numerals Example - Successor

$$0 \equiv \lambda f.\lambda x.x$$

$$1 \equiv \lambda f.\lambda x.f\ x$$

$$\mathit{successor} \equiv \lambda n.\lambda f.\lambda x.f\ (n\ f\ x)$$

$$\begin{aligned} & \mathit{successor}\ 1 \\ \equiv\ & (\lambda n.\lambda f.\lambda x.f\ (n\ f\ x))\ 1 \\ \to\ & \lambda f.\lambda x.f\ (1\ f\ x) \\ \equiv\ & \lambda f.\lambda x.f\ ((\lambda f.\lambda x.f\ x)\ f\ x) \\ \to\ & \lambda f.\lambda x.f\ ((\lambda x.f\ x)\ x) \end{aligned}$$

Note
We use *Normal Order Reduction* to reduce inside abstractions!

Numerals Example - Successor

$$
\begin{aligned}
0 &\equiv \lambda f.\lambda x.x \\
1 &\equiv \lambda f.\lambda x.f\ x \\
\\
\mathit{successor} &\equiv \lambda n.\lambda f.\lambda x.f\ (n\ f\ x)
\end{aligned}
$$

$$
\begin{aligned}
& \mathit{successor}\ 1 \\
\equiv\ & (\lambda n.\lambda f.\lambda x.f\ (n\ f\ x))\ 1 \\
\rightarrow\ & \lambda f.\lambda x.f\ (1\ f\ x) \\
\equiv\ & \lambda f.\lambda x.f\ ((\lambda f.\lambda x.f\ x)\ f\ x) \\
\rightarrow\ & \lambda f.\lambda x.f\ ((\lambda x.f\ x)\ x) \\
\rightarrow\ & \lambda f.\lambda x.f\ (f\ x)
\end{aligned}
$$

Note
We use *Normal Order Reduction* to reduce inside abstractions!

Numerals Example - Successor

$$
\begin{aligned}
0 &\equiv \lambda f.\lambda x.x \\
1 &\equiv \lambda f.\lambda x.f\ x \\
\\
\mathit{successor} &\equiv \lambda n.\lambda f.\lambda x.f\ (n\ f\ x)
\end{aligned}
$$

$$
\begin{aligned}
& \mathit{successor}\ 1 \\
\equiv\ & (\lambda n.\lambda f.\lambda x.f\ (n\ f\ x))\ 1 \\
\rightarrow\ & \lambda f.\lambda x.f\ (1\ f\ x) \\
\equiv\ & \lambda f.\lambda x.f\ ((\lambda f.\lambda x.f\ x)\ f\ x) \\
\rightarrow\ & \lambda f.\lambda x.f\ ((\lambda x.f\ x)\ x) \\
\rightarrow\ & \lambda f.\lambda x.f\ (f\ x) \\
\equiv\ & 2
\end{aligned}
$$

Note
We use *Normal Order Reduction* to reduce inside abstractions!

Numerals Example - 0 + 0

$$0 \equiv \qquad \lambda f.\lambda x.x$$

Numerals Example - 0 + 0

$$0 \equiv \lambda f.\lambda x.x$$

$$plus \equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x)$$

Numerals Example - 0 + 0

$plus\ 0\ 0$

$$\begin{aligned} 0 &\equiv \lambda f.\lambda x.x \\ plus &\equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x) \end{aligned}$$

Numerals Example - 0 + 0

plus 0 0

$$
\begin{aligned}
0 &\equiv \lambda f.\lambda x.x \\
plus &\equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x)
\end{aligned}
$$

Numerals Example - 0 + 0

$$
\begin{aligned}
& \textit{plus}\ 0\ 0 \\
\equiv\ & (\lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x))\ 0\ 0
\end{aligned}
$$

$$
\begin{aligned}
0 &\equiv \lambda f.\lambda x.x \\
\textit{plus} &\equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x)
\end{aligned}
$$

Numerals Example - 0 + 0

$$\begin{aligned} & plus\ 0\ 0 \\ \equiv\ & (\lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x))\ 0\ 0 \end{aligned}$$

$$0 \equiv \lambda f.\lambda x.x$$

$$plus \equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x)$$

Numerals Example - 0 + 0

$$
\begin{aligned}
& \mathit{plus}\ 0\ 0 \\
\equiv\quad & (\lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x))\ 0\ 0 \\
\rightarrow\quad & (\lambda n.\lambda f.\lambda x.0\ f\ (n\ f\ x))\ 0
\end{aligned}
$$

$$0 \equiv \lambda f.\lambda x.x$$

$$\mathit{plus} \equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x)$$

Numerals Example - 0 + 0

$$
\begin{aligned}
& \mathit{plus}\ 0\ 0 \\
\equiv\ & (\lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x))\ 0\ 0 \\
\rightarrow\ & (\lambda n.\lambda f.\lambda x.0\ f\ (n\ f\ x))\ 0
\end{aligned}
$$

$$0 \equiv \lambda f.\lambda x.x$$

$$\mathit{plus} \equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x)$$

Numerals Example - 0 + 0

$$\begin{aligned} & \mathit{plus}\ 0\ 0 \\ \equiv\ & (\lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x))\ 0\ 0 \\ \rightarrow\ & (\lambda n.\lambda f.\lambda x.0\ f\ (n\ f\ x))\ 0 \\ \rightarrow\ & \lambda f.\lambda x.0\ f\ (0\ f\ x) \end{aligned}$$

$$0 \equiv \lambda f.\lambda x.x$$

$$\mathit{plus} \equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x)$$

Numerals Example - 0 + 0

$$0 \equiv \lambda f.\lambda x.x$$

$$plus \equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x)$$

$$\begin{aligned} & plus\ 0\ 0 \\ \equiv\ & (\lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x))\ 0\ 0 \\ \rightarrow\ & (\lambda n.\lambda f.\lambda x.0\ f\ (n\ f\ x))\ 0 \\ \rightarrow\ & \lambda f.\lambda x.0\ f\ (0\ f\ x) \end{aligned}$$

Numerals Example - 0 + 0

$$\begin{aligned} & & plus\ 0\ 0 \\ & \equiv & (\lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x))\ 0\ 0 \\ & \to & (\lambda n.\lambda f.\lambda x.0\ f\ (n\ f\ x))\ 0 \\ & \to & \lambda f.\lambda x.0\ f\ (0\ f\ x) \\ 0 \equiv \quad \lambda f.\lambda x.x & \equiv & \lambda f.\lambda x.(\lambda f.\lambda x.x)\ f\ (0\ f\ x) \end{aligned}$$

$$plus \equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x)$$

Numerals Example - 0 + 0

$$\begin{aligned} & \mathit{plus}\ 0\ 0 \\ \equiv\ & (\lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x))\ 0\ 0 \\ \to\ & (\lambda n.\lambda f.\lambda x.0\ f\ (n\ f\ x))\ 0 \\ \to\ & \lambda f.\lambda x.0\ f\ (0\ f\ x) \\ \equiv\ & \lambda f.\lambda x.(\lambda f.\lambda x.x)\ f\ (0\ f\ x) \end{aligned}$$

$$0 \equiv \lambda f.\lambda x.x$$

$$\mathit{plus} \equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x)$$

Numerals Example - 0 + 0

$$0 \equiv \lambda f.\lambda x.x$$

$$plus \equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x)$$

$$\begin{aligned}
& plus\ 0\ 0 \\
\equiv\ & (\lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x))\ 0\ 0 \\
\rightarrow\ & (\lambda n.\lambda f.\lambda x.0\ f\ (n\ f\ x))\ 0 \\
\rightarrow\ & \lambda f.\lambda x.0\ f\ (0\ f\ x) \\
\equiv\ & \lambda f.\lambda x.(\lambda f.\lambda x.x)\ f\ (0\ f\ x) \\
\rightarrow\ & \lambda f.\lambda x.(\lambda x.x)\ (0\ f\ x)
\end{aligned}$$

Numerals Example - 0 + 0

$$0 \equiv \lambda f.\lambda x.x$$

$$plus \equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x)$$

$$\begin{aligned}
& plus\ 0\ 0 \\
\equiv\ & (\lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x))\ 0\ 0 \\
\rightarrow\ & (\lambda n.\lambda f.\lambda x.0\ f\ (n\ f\ x))\ 0 \\
\rightarrow\ & \lambda f.\lambda x.0\ f\ (0\ f\ x) \\
\equiv\ & \lambda f.\lambda x.(\lambda f.\lambda x.x)\ f\ (0\ f\ x) \\
\rightarrow\ & \lambda f.\lambda x.(\lambda x.x)\ (0\ f\ x)
\end{aligned}$$

Numerals Example - 0 + 0

$$0 \equiv \lambda f.\lambda x.x$$

$$plus \equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x)$$

$$\begin{aligned}
& plus\ 0\ 0 \\
\equiv\ & (\lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x))\ 0\ 0 \\
\rightarrow\ & (\lambda n.\lambda f.\lambda x.0\ f\ (n\ f\ x))\ 0 \\
\rightarrow\ & \lambda f.\lambda x.0\ f\ (0\ f\ x) \\
\equiv\ & \lambda f.\lambda x.(\lambda f.\lambda x.x)\ f\ (0\ f\ x) \\
\rightarrow\ & \lambda f.\lambda x.(\lambda x.x)\ (0\ f\ x) \\
\rightarrow\ & \lambda f.\lambda x.0\ f\ x
\end{aligned}$$

Numerals Example - 0 + 0

$$0 \equiv \lambda f.\lambda x.x$$

$$plus \equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x)$$

$$\begin{aligned}
& plus\ 0\ 0 \\
\equiv\ & (\lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x))\ 0\ 0 \\
\rightarrow\ & (\lambda n.\lambda f.\lambda x.0\ f\ (n\ f\ x))\ 0 \\
\rightarrow\ & \lambda f.\lambda x.0\ f\ (0\ f\ x) \\
\equiv\ & \lambda f.\lambda x.(\lambda f.\lambda x.x)\ f\ (0\ f\ x) \\
\rightarrow\ & \lambda f.\lambda x.(\lambda x.x)\ (0\ f\ x) \\
\rightarrow\ & \lambda f.\lambda x.0\ f\ x
\end{aligned}$$

Numerals Example - 0 + 0

$$0 \equiv \lambda f.\lambda x.x$$

$$plus \equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x)$$

$$\begin{aligned}
& plus\ 0\ 0 \\
\equiv\ & (\lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x))\ 0\ 0 \\
\to\ & (\lambda n.\lambda f.\lambda x.0\ f\ (n\ f\ x))\ 0 \\
\to\ & \lambda f.\lambda x.0\ f\ (0\ f\ x) \\
\equiv\ & \lambda f.\lambda x.(\lambda f.\lambda x.x)\ f\ (0\ f\ x) \\
\to\ & \lambda f.\lambda x.(\lambda x.x)\ (0\ f\ x) \\
\to\ & \lambda f.\lambda x.0\ f\ x \\
\equiv\ & \lambda f.\lambda x.(\lambda f.\lambda x.x)\ f\ x
\end{aligned}$$

Numerals Example - 0 + 0

$$0 \equiv \lambda f.\lambda x.x$$

$$plus \equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x)$$

$$\begin{aligned}
& plus\ 0\ 0 \\
\equiv\ & (\lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x))\ 0\ 0 \\
\rightarrow\ & (\lambda n.\lambda f.\lambda x.0\ f\ (n\ f\ x))\ 0 \\
\rightarrow\ & \lambda f.\lambda x.0\ f\ (0\ f\ x) \\
\equiv\ & \lambda f.\lambda x.(\lambda f.\lambda x.x)\ f\ (0\ f\ x) \\
\rightarrow\ & \lambda f.\lambda x.(\lambda x.x)\ (0\ f\ x) \\
\rightarrow\ & \lambda f.\lambda x.0\ f\ x \\
\equiv\ & \lambda f.\lambda x.(\lambda f.\lambda x.x)\ f\ x
\end{aligned}$$

Numerals Example - 0 + 0

$$0 \equiv \lambda f.\lambda x.x$$

$$plus \equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x)$$

$$\begin{aligned}
& plus\ 0\ 0 \\
\equiv\ & (\lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x))\ 0\ 0 \\
\rightarrow\ & (\lambda n.\lambda f.\lambda x.0\ f\ (n\ f\ x))\ 0 \\
\rightarrow\ & \lambda f.\lambda x.0\ f\ (0\ f\ x) \\
\equiv\ & \lambda f.\lambda x.(\lambda f.\lambda x.x)\ f\ (0\ f\ x) \\
\rightarrow\ & \lambda f.\lambda x.(\lambda x.x)\ (0\ f\ x) \\
\rightarrow\ & \lambda f.\lambda x.0\ f\ x \\
\equiv\ & \lambda f.\lambda x.(\lambda f.\lambda x.x)\ f\ x \\
\rightarrow\ & \lambda f.\lambda x.(\lambda x.x)\ x
\end{aligned}$$

Numerals Example - 0 + 0

$$0 \equiv \lambda f.\lambda x.x$$

$$plus \equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x)$$

$$
\begin{aligned}
& plus\ 0\ 0 \\
\equiv\ & (\lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x))\ 0\ 0 \\
\to\ & (\lambda n.\lambda f.\lambda x.0\ f\ (n\ f\ x))\ 0 \\
\to\ & \lambda f.\lambda x.0\ f\ (0\ f\ x) \\
\equiv\ & \lambda f.\lambda x.(\lambda f.\lambda x.x)\ f\ (0\ f\ x) \\
\to\ & \lambda f.\lambda x.(\lambda x.x)\ (0\ f\ x) \\
\to\ & \lambda f.\lambda x.0\ f\ x \\
\equiv\ & \lambda f.\lambda x.(\lambda f.\lambda x.x)\ f\ x \\
\to\ & \lambda f.\lambda x.(\lambda x.x)\ x
\end{aligned}
$$

Numerals Example - 0 + 0

$$0 \equiv \lambda f.\lambda x.x$$

$$plus \equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x)$$

$$\begin{aligned}
& plus\ 0\ 0 \\
\equiv\ & (\lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x))\ 0\ 0 \\
\rightarrow\ & (\lambda n.\lambda f.\lambda x.0\ f\ (n\ f\ x))\ 0 \\
\rightarrow\ & \lambda f.\lambda x.0\ f\ (0\ f\ x) \\
\equiv\ & \lambda f.\lambda x.(\lambda f.\lambda x.x)\ f\ (0\ f\ x) \\
\rightarrow\ & \lambda f.\lambda x.(\lambda x.x)\ (0\ f\ x) \\
\rightarrow\ & \lambda f.\lambda x.0\ f\ x \\
\equiv\ & \lambda f.\lambda x.(\lambda f.\lambda x.x)\ f\ x \\
\rightarrow\ & \lambda f.\lambda x.(\lambda x.x)\ x \\
\rightarrow\ & \lambda f.\lambda x.x
\end{aligned}$$

Numerals Example - 0 + 0

$$0 \equiv \lambda f.\lambda x.x$$

$$plus \equiv \lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x)$$

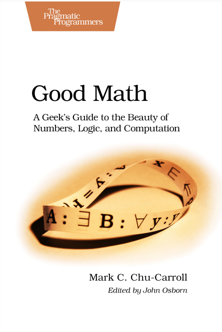
$$\begin{aligned}
& plus\ 0\ 0 \\
\equiv\ & (\lambda m.\lambda n.\lambda f.\lambda x.m\ f\ (n\ f\ x))\ 0\ 0 \\
\rightarrow\ & (\lambda n.\lambda f.\lambda x.0\ f\ (n\ f\ x))\ 0 \\
\rightarrow\ & \lambda f.\lambda x.0\ f\ (0\ f\ x) \\
\equiv\ & \lambda f.\lambda x.(\lambda f.\lambda x.x)\ f\ (0\ f\ x) \\
\rightarrow\ & \lambda f.\lambda x.(\lambda x.x)\ (0\ f\ x) \\
\rightarrow\ & \lambda f.\lambda x.0\ f\ x \\
\equiv\ & \lambda f.\lambda x.(\lambda f.\lambda x.x)\ f\ x \\
\rightarrow\ & \lambda f.\lambda x.(\lambda x.x)\ x \\
\rightarrow\ & \lambda f.\lambda x.x \\
\equiv\ & 0
\end{aligned}$$

End

Thanks

- Hope you enjoyed this talk and learned something new.
- Hope it wasn't too much math and dusty formulas ... :)

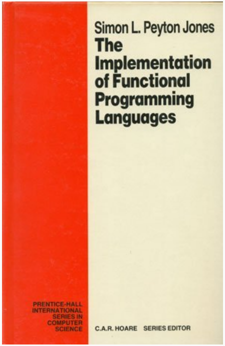
Good Math



"A Geek's Guide to the Beauty of Numbers, Logic, and Computation"

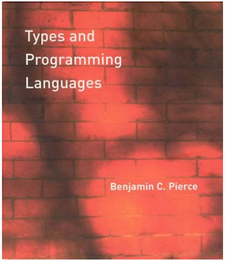
- Easy to understand

The Implementation of Functional Programming Languages



- How to compile to the Lambda Calculus?
- Out-of-print, but freely available
  - https://www.microsoft.com/en-us/research/publication/the-implementation-of-functional-programming-languages/

Types and Programming Languages



- Types systems explained by building interpreters and proving properties
- Very "mathematical", but very complete and self-contained