

# Функциональное программирование vs. Интуиционистская логика

Лозов Пётр


СПбГУ

28 ноября 2019 г.

# Наш план

1. Соответствие Карри — Ховарда
2. Призраки минувших доказательств

Разработано Алонзо Чёрчем для формализации и анализа понятия вычислимости в 1930 годах

 [An unsolvable problem of elementary number theory](#)  
Church, A.  
1936

## Синтаксис $\lambda$ -исчисления

$$X = \{a, b, c, x, y, z, \dots\}$$

$$\begin{array}{lcl} S & = & X \quad [Var] \\ & | & \lambda X. S \quad [Abst] \\ & | & S S \quad [App] \end{array}$$

## Синтаксис $\lambda$ -исчисления

$$X = \{a, b, c, x, y, z, \dots\}$$

$$\begin{array}{lcl} S & = & X \quad [Var] \\ & | & \lambda X. S \quad [Abst] \\ & | & S S \quad [App] \end{array}$$

## Примеры

- $x$
- $\lambda z. z$
- $(\lambda x. x x)(\lambda x. x x)$

Основное правило преобразования  $\lambda$ -выражения

$$(\lambda x. A) B \rightsquigarrow^\beta A[x \leftarrow B]$$

Основное правило преобразования  $\lambda$ -выражения

$$(\lambda x. A) B \rightsquigarrow^{\beta} A[x \leftarrow B]$$

Примеры

- $(\lambda z. z) x$

Основное правило преобразования  $\lambda$ -выражения

$$(\lambda x. A) B \rightsquigarrow^\beta A[x \leftarrow B]$$

Примеры

- $(\lambda z. z) x \rightsquigarrow^\beta z[z \leftarrow x] \rightsquigarrow x$



Основное правило преобразования  $\lambda$ -выражения

$$(\lambda x. A) B \rightsquigarrow^{\beta} A[x \leftarrow B]$$

Примеры

- $(\lambda z. z) x \rightsquigarrow^{\beta} z[z \leftarrow x] \rightsquigarrow x$
- $(\lambda x. \lambda x. x) a$

Основное правило преобразования  $\lambda$ -выражения

$$(\lambda x. A) B \rightsquigarrow^\beta A[x \leftarrow B]$$

Примеры

- $(\lambda z. z) x \rightsquigarrow^\beta z[z \leftarrow x] \rightsquigarrow x$
- $(\lambda x. \lambda x. x) a \rightsquigarrow^\beta (\lambda x. x)[x \leftarrow a]$

Основное правило преобразования  $\lambda$ -выражения

$$(\lambda x. A) B \rightsquigarrow^\beta A[x \leftarrow B]$$

Примеры

- $(\lambda z. z) x \rightsquigarrow^\beta z[z \leftarrow x] \rightsquigarrow x$
- $(\lambda x. \lambda x. x) a \rightsquigarrow^\beta (\lambda x. x)[x \leftarrow a] \rightsquigarrow \lambda x. x$

Основное правило преобразования  $\lambda$ -выражения

$$(\lambda x. A) B \rightsquigarrow^\beta A[x \leftarrow B]$$

Примеры

- $(\lambda z. z) x \rightsquigarrow^\beta z[z \leftarrow x] \rightsquigarrow x$
- $(\lambda x. \lambda x. x) a \rightsquigarrow^\beta (\lambda x. x)[x \leftarrow a] \rightsquigarrow \lambda x. x$
- $(\lambda z. \lambda y. z y) y$

## Основное правило преобразования $\lambda$ -выражения

$$(\lambda x. A) B \rightsquigarrow^\beta A[x \leftarrow B]$$

### Примеры

- $(\lambda z. z) x \rightsquigarrow^\beta z[z \leftarrow x] \rightsquigarrow x$
- $(\lambda x. \lambda x. x) a \rightsquigarrow^\beta (\lambda x. x)[x \leftarrow a] \rightsquigarrow \lambda x. x$
- $(\lambda z. \lambda y. z y) y \rightsquigarrow^\beta (\lambda y. z y)[z \leftarrow y]$

## Основное правило преобразования $\lambda$ -выражения

$$(\lambda x. A) B \rightsquigarrow^\beta A[x \leftarrow B]$$

### Примеры

- $(\lambda z. z) x \rightsquigarrow^\beta z[z \leftarrow x] \rightsquigarrow x$
- $(\lambda x. \lambda x. x) a \rightsquigarrow^\beta (\lambda x. x)[x \leftarrow a] \rightsquigarrow \lambda x. x$
- $(\lambda z. \lambda y. z y) y \rightsquigarrow^\beta (\lambda y. z y)[z \leftarrow y] \rightsquigarrow^\alpha (\lambda t. z t)[z \leftarrow y]$

## Основное правило преобразования $\lambda$ -выражения

$$(\lambda x. A) B \rightsquigarrow^\beta A[x \leftarrow B]$$

### Примеры

- $(\lambda z. z) x \rightsquigarrow^\beta z[z \leftarrow x] \rightsquigarrow x$
- $(\lambda x. \lambda x. x) a \rightsquigarrow^\beta (\lambda x. x)[x \leftarrow a] \rightsquigarrow \lambda x. x$
- $(\lambda z. \lambda y. z y) y \rightsquigarrow^\beta (\lambda y. z y)[z \leftarrow y] \rightsquigarrow^\alpha (\lambda t. z t)[z \leftarrow y] \rightsquigarrow \lambda t. y t$

- Выразимы натуральные числа
- Выразима арифметические операции
- Выразим условный оператор
- Выразима рекурсия
- Полнота по Тьюрингу



- Отсутствие нормальной формы у некоторых выражений

- Отсутствие нормальной формы у некоторых выражений  
 $(\lambda x. x x)(\lambda x. x x)$

# Недостатки $\lambda$ -исчисления

- Отсутствие нормальной формы у некоторых выражений  
 $(\lambda x. x x)(\lambda x. x x) \rightsquigarrow^\beta (\lambda x. x x)[x. \leftarrow \lambda x. x x]$

# Недостатки $\lambda$ -исчисления

- Отсутствие нормальной формы у некоторых выражений

$$(\lambda x. x x)(\lambda x. x x) \rightsquigarrow^{\beta} (\lambda x. x x)[x. \leftarrow \lambda x. x x] \rightsquigarrow (\lambda x. x x)(\lambda x. x x)$$

# Недостатки $\lambda$ -исчисления

- Отсутствие нормальной формы у некоторых выражений

$$(\lambda x. x x)(\lambda x. x x) \rightsquigarrow^\beta (\lambda x. x x)[x. \leftarrow \lambda x. x x] \rightsquigarrow (\lambda x. x x)(\lambda x. x x) \rightsquigarrow^\beta \dots$$

- Важен порядок редукции

# Недостатки $\lambda$ -исчисления

- Отсутствие нормальной формы у некоторых выражений

$$(\lambda x. x x)(\lambda x. x x) \rightsquigarrow^\beta (\lambda x. x x)[x. \leftarrow \lambda x. x x] \rightsquigarrow (\lambda x. x x)(\lambda x. x x) \rightsquigarrow^\beta \dots$$

- Важен порядок редукции

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

# Недостатки $\lambda$ -исчисления

- Отсутствие нормальной формы у некоторых выражений

$$(\lambda x. x x)(\lambda x. x x) \rightsquigarrow^\beta (\lambda x. x x)[x. \leftarrow \lambda x. x x] \rightsquigarrow (\lambda x. x x)(\lambda x. x x) \rightsquigarrow^\beta \dots$$

- Важен порядок редукции

$$(\lambda x. a)((\lambda x. x x)(\lambda x. x x)) \rightsquigarrow^\beta (\lambda x. a)((\lambda x. x x)[x. \leftarrow \lambda x. x x]) \rightsquigarrow^\beta \dots$$

# Недостатки $\lambda$ -исчисления

- Отсутствие нормальной формы у некоторых выражений

$$(\lambda x. x x)(\lambda x. x x) \rightsquigarrow^\beta (\lambda x. x x)[x. \leftarrow \lambda x. x x] \rightsquigarrow (\lambda x. x x)(\lambda x. x x) \rightsquigarrow^\beta \dots$$

- Важен порядок редукции

$$(\lambda x. a)((\lambda x. x x)(\lambda x. x x)) \rightsquigarrow^\beta (\lambda x. a)((\lambda x. x x)[x. \leftarrow \lambda x. x x]) \rightsquigarrow^\beta \dots$$

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



# Недостатки $\lambda$ -исчисления

- Отсутствие нормальной формы у некоторых выражений

$$(\lambda x. x x)(\lambda x. x x) \rightsquigarrow^\beta (\lambda x. x x)[x. \leftarrow \lambda x. x x] \rightsquigarrow (\lambda x. x x)(\lambda x. x x) \rightsquigarrow^\beta \dots$$

- Важен порядок редукции

$$(\lambda x. a)((\lambda x. x x)(\lambda x. x x)) \rightsquigarrow^\beta (\lambda x. a)((\lambda x. x x)[x. \leftarrow \lambda x. x x]) \rightsquigarrow^\beta \dots$$

$$(\lambda x. a)((\lambda x. x x)(\lambda x. x x)) \rightsquigarrow^\beta a[x \leftarrow (\lambda x. x x)(\lambda x. x x)]$$

# Недостатки $\lambda$ -исчисления

- Отсутствие нормальной формы у некоторых выражений

$$(\lambda x. x x)(\lambda x. x x) \rightsquigarrow^\beta (\lambda x. x x)[x. \leftarrow \lambda x. x x] \rightsquigarrow (\lambda x. x x)(\lambda x. x x) \rightsquigarrow^\beta \dots$$

- Важен порядок редукции


$$(\lambda x. a)((\lambda x. x x)(\lambda x. x x)) \rightsquigarrow^\beta (\lambda x. a)((\lambda x. x x)[x. \leftarrow \lambda x. x x]) \rightsquigarrow^\beta \dots$$

$$(\lambda x. a)((\lambda x. x x)(\lambda x. x x)) \rightsquigarrow^\beta a[x \leftarrow (\lambda x. x x)(\lambda x. x x)] \rightsquigarrow a$$

- Полнота по Тьюрингу

# Просто типизированное $\lambda$ -исчисление (STLC)

Введено Алонзо Чёрчем для исключения  $\lambda$ -выражений с “плохим” поведением

 [A Formulation of the Simple Theory of Types](#)  
Church, A.  
1940

## Синтаксис типов

$$V = \{\alpha, \beta, \gamma, \dots\}$$

$$T = V \mid T \rightarrow T$$

## Синтаксис типов

$$V = \{\alpha, \beta, \gamma, \dots\}$$

$$T = V \mid T \rightarrow T$$

## Примеры

- $\alpha$
- $\beta \rightarrow \beta$
- $\alpha \rightarrow (\alpha \rightarrow \beta) \rightarrow \beta$

Множество контекстов:

$$S_\Gamma = \Lambda \mid V : T, S_\Gamma$$

Множество контекстов:

$$S_{\Gamma} = \Lambda \mid V : T, S_{\Gamma}$$

Пример

$$\Gamma = x : \alpha, y : \beta \rightarrow \gamma, \Lambda$$

В контексте  $\Gamma$   $\lambda$ -выражение  $A$  имеет тип  $t$

$$\Gamma \vdash A : t$$



В контексте  $\Gamma$   $\lambda$ -выражение  $A$  имеет тип  $t$

$$\Gamma \vdash A : t$$

$\lambda$ -выражение  $A$  имеет тип  $t$

$$\Lambda \vdash A : t$$

$$\frac{x : t \in \Gamma}{\Gamma \vdash x : t} \text{ (Var)}$$

$$\frac{x : t_1, \Gamma \vdash A : t_2}{\Gamma \vdash \lambda x. A : t_1 \rightarrow t_2} \text{ (Abst)}$$

$$\frac{\Gamma \vdash A : t_0 \rightarrow t \quad \Gamma \vdash B : t_0}{\Gamma \vdash AB : t} \text{ (App)}$$

## Пример вывода (1)

$$\Lambda \vdash (\lambda x. x)(\lambda y. y) : t$$

## Пример вывода (2)

$$\frac{\Lambda \vdash \lambda x.x : t_0 \rightarrow t \qquad \Lambda \vdash \lambda y.y : t_0}{\Lambda \vdash (\lambda x.x)(\lambda y.y) : t} \text{ (App)}$$

## Пример вывода (3)

$$\frac{\frac{x : t_0, \Lambda \vdash x : t}{\Lambda \vdash \lambda x. x : t_0 \rightarrow t} \text{ (Abst)} \quad \Lambda \vdash \lambda y. y : t_0}{\Lambda \vdash (\lambda x. x)(\lambda y. y) : t} \text{ (App)}$$

## Пример вывода (4)

$$\frac{\frac{\frac{x : t_0 \in x : t, \Lambda}{x : t_0, \Lambda \vdash x : t} \text{ (Var)}}{\Lambda \vdash \lambda x. x : t_0 \rightarrow t} \text{ (Abst)} \quad \Lambda \vdash \lambda y. y : t_0}{\Lambda \vdash (\lambda x. x)(\lambda y. y) : t} \text{ (App)}$$

## Пример вывода (5)

$$\frac{\frac{\frac{x : t \in x : t, \Lambda}{x : t, \Lambda \vdash x : t} \text{ (Var)}}{\Lambda \vdash \lambda x. x : t \rightarrow t} \text{ (Abst)} \quad \Lambda \vdash \lambda y. y : t}{\Lambda \vdash (\lambda x. x)(\lambda y. y) : t} \text{ (App)}$$

## Пример вывода (6)

$$\frac{\frac{\frac{x : t_1 \rightarrow t_2 \in x : t_1 \rightarrow t_2, \Lambda}{x : t_1 \rightarrow t_2, \Lambda \vdash x : t_1 \rightarrow t_2} \text{ (Var)}}{\Lambda \vdash \lambda x. x : (t_1 \rightarrow t_2) \rightarrow t_1 \rightarrow t_2} \text{ (Abst)} \quad \Lambda \vdash \lambda y. y : t_1 \rightarrow t_2}{\Lambda \vdash (\lambda x. x)(\lambda y. y) : t_1 \rightarrow t_2} \text{ (App)}$$



## Пример вывода (7)

$$\frac{\frac{\frac{x : t_1 \rightarrow t_2 \in x : t_1 \rightarrow t_2, \Lambda}{x : t_1 \rightarrow t_2, \Lambda \vdash x : t_1 \rightarrow t_2} \text{ (Var)}}{\Lambda \vdash \lambda x. x : (t_1 \rightarrow t_2) \rightarrow t_1 \rightarrow t_2} \text{ (Abst)} \quad \frac{y : t_1, \Lambda \vdash y : t_2}{\Lambda \vdash \lambda y. y : t_1 \rightarrow t_2} \text{ (Abst)}}{\Lambda \vdash (\lambda x. x)(\lambda y. y) : t_1 \rightarrow t_2} \text{ (App)}$$

## Пример вывода (8)

$$\frac{\frac{\frac{x : t_1 \rightarrow t_2 \in x : t_1 \rightarrow t_2, \Lambda}{x : t_1 \rightarrow t_2, \Lambda \vdash x : t_1 \rightarrow t_2} \text{ (Var)}}{\Lambda \vdash \lambda x. x : (t_1 \rightarrow t_2) \rightarrow t_1 \rightarrow t_2} \text{ (Abst)} \quad \frac{\frac{\frac{y : t_2 \in y : t_1, \Lambda}{y : t_1, \Lambda \vdash y : t_2} \text{ (Var)}}{\Lambda \vdash \lambda y. y : t_1 \rightarrow t_2} \text{ (Abst)}}{\Lambda \vdash (\lambda x. x)(\lambda y. y) : t_1 \rightarrow t_2} \text{ (App)}$$

## Пример вывода (9)

$$\frac{\frac{\frac{x : t_1 \rightarrow t_1 \in x : t_1 \rightarrow t_1, \Lambda}{x : t_1 \rightarrow t_1, \Lambda \vdash x : t_1 \rightarrow t_1} \text{ (Var)}}{\Lambda \vdash \lambda x. x : (t_1 \rightarrow t_1) \rightarrow t_1 \rightarrow t_1} \text{ (Abst)} \quad \frac{\frac{\frac{y : t_1 \in y : t_1, \Lambda}{y : t_1, \Lambda \vdash y : t_1} \text{ (Var)}}{\Lambda \vdash \lambda y. y : t_1 \rightarrow t_1} \text{ (Abst)}}{\Lambda \vdash (\lambda x. x)(\lambda y. y) : t_1 \rightarrow t_1} \text{ (App)}$$

## Пример вывода (10)

$$\frac{\frac{\frac{x : \alpha \rightarrow \alpha \in x : \alpha \rightarrow \alpha, \Lambda}{x : \alpha \rightarrow \alpha, \Lambda \vdash x : \alpha \rightarrow \alpha} \text{ (Var)}}{\Lambda \vdash \lambda x. x : (\alpha \rightarrow \alpha) \rightarrow \alpha \rightarrow \alpha} \text{ (Abst)} \quad \frac{\frac{\frac{y : \alpha \in y : \alpha, \Lambda}{y : \alpha, \Lambda \vdash y : \alpha} \text{ (Var)}}{\Lambda \vdash \lambda y. y : \alpha \rightarrow \alpha} \text{ (Abst)}}{\Lambda \vdash (\lambda x. x)(\lambda y. y) : \alpha \rightarrow \alpha} \text{ (App)}$$

- Если выражение имеет тип, то оно имеет нормальную форму
- Если выражение имеет тип, то порядок редукции не важен
- Свойство безопасности: сохранение типа после  $\beta$ -редукции
- Нет полноты по Тьюрингу

Интуиционистская логика - раздел современной математической логики, отражающий идеи конструктивного доказательства

- Вместо истины и ложности — доказуемость
- Нет закона исключённого третьего
- Нет закона снятия двойного отрицания

## Синтаксис формул

$$V = \{\alpha, \beta, \gamma, \dots\}$$

$$\Phi = \perp \mid V \mid \Phi \vee \Phi \mid \Phi \wedge \Phi \mid \Phi \rightarrow \Phi$$

## Обозначения

$$\neg\phi = \phi \rightarrow \perp$$

$$\top = \perp \rightarrow \perp$$

Список гипотез

$$\Gamma = \Phi, \Gamma \mid \Lambda$$

Суждение (из списка гипотез  $\Gamma$  следует формула  $A$ )

$$\Gamma \vdash A$$



## Часть правил натурального вывода

$$\frac{\phi \in \Gamma}{\Gamma \vdash \phi} \text{ (Ax)}$$

$$\frac{\phi, \Gamma \vdash \psi}{\Gamma \vdash \phi \rightarrow \psi} \text{ (Impl I)}$$

$$\frac{\phi, \Gamma \vdash \phi \rightarrow \psi \quad \phi, \Gamma \vdash \psi}{\Gamma \vdash \phi} \text{ (Impl E)}$$

## Пример вывода

$$\frac{\frac{\frac{\alpha \rightarrow \beta \in \alpha, \alpha \rightarrow \beta, \Lambda}{\alpha, \alpha \rightarrow \beta, \Lambda \vdash \alpha \rightarrow \beta} \text{ (Ax)} \quad \frac{\alpha \in \alpha, \alpha \rightarrow \beta, \Lambda}{\alpha, \alpha \rightarrow \beta, \Lambda \vdash \alpha} \text{ (Ax)}}{\alpha, \alpha \rightarrow \beta, \Lambda \vdash \beta} \text{ (Impl E)} \\ \frac{\alpha, \alpha \rightarrow \beta, \Lambda \vdash \beta}{\alpha, \Lambda \vdash (\alpha \rightarrow \beta) \rightarrow \beta} \text{ (Impl I)} \\ \frac{\alpha, \Lambda \vdash (\alpha \rightarrow \beta) \rightarrow \beta}{\Lambda \vdash \alpha \rightarrow (\alpha \rightarrow \beta) \rightarrow \beta} \text{ (Impl I)}$$

$$(Ax) \frac{\phi \in \Gamma}{\Gamma \vdash \phi}$$

$$\frac{x : t \in \Gamma}{\Gamma \vdash x : t} (Var)$$

$$(Impl I) \frac{\phi, \Gamma \vdash \psi}{\Gamma \vdash \phi \rightarrow \psi}$$

$$\frac{x : t_1, \Gamma \vdash A : t_2}{\Gamma \vdash \lambda x. A : t_1 \rightarrow t_2} (Abst)$$

$$(Impl E) \frac{\phi, \Gamma \vdash \phi \rightarrow \psi \quad \phi, \Gamma \vdash \psi}{\Gamma \vdash \psi}$$

$$\frac{\Gamma \vdash A : t_0 \rightarrow t \quad \Gamma \vdash B : t_0}{\Gamma \vdash AB : t} (App)$$

# Правила натурального вывода для конъюнкции

$$(\text{Conj I}) \frac{\Gamma \vdash \phi \quad \Gamma \vdash \psi}{\Gamma \vdash \phi \wedge \psi}$$

$$(\text{Conj E1}) \frac{\Gamma \vdash \phi \wedge \psi}{\Gamma \vdash \phi}$$

$$(\text{Conj E2}) \frac{\Gamma \vdash \phi \wedge \psi}{\Gamma \vdash \psi}$$

## Сравним (2)

$$(\text{Conj I}) \frac{\Gamma \vdash \phi \quad \Gamma \vdash \psi}{\Gamma \vdash \phi \wedge \psi}$$

$$\frac{\Gamma \vdash A : t_1 \quad \Gamma \vdash B : t_2}{\Gamma \vdash (A, B) : (t_1, t_2)} (\text{Pair})$$

$$(\text{Conj E1}) \frac{\Gamma \vdash \phi \wedge \psi}{\Gamma \vdash \phi}$$

$$\frac{\Gamma \vdash A : (t_1, t_2)}{\Gamma \vdash \text{fst } A : t_1} (\text{Fst})$$

$$(\text{Conj E2}) \frac{\Gamma \vdash \phi \wedge \psi}{\Gamma \vdash \psi}$$

$$\frac{\Gamma \vdash A : (t_1, t_2)}{\Gamma \vdash \text{snd } A : t_2} (\text{Snd})$$

# Правила натурального вывода для дизъюнкции

$$(\text{Disj I1}) \frac{\Gamma \vdash \phi}{\Gamma \vdash \phi \vee \psi}$$

$$(\text{Disj I2}) \frac{\Gamma \vdash \psi}{\Gamma \vdash \phi \vee \psi}$$

$$(\text{Disj E}) \frac{\Gamma \vdash \phi \vee \psi \quad \phi, \Gamma \vdash \nu \quad \psi, \Gamma \vdash \nu}{\Gamma \vdash \nu}$$

## Сравним (3)

$$\text{(Disj I1)} \frac{\Gamma \vdash \phi}{\Gamma \vdash \phi \vee \psi}$$

$$\frac{\Gamma \vdash A : t}{\Gamma \vdash \textit{Left } A : \textit{Either } t \ t_0} \text{ (Left)}$$

$$\text{(Disj I2)} \frac{\Gamma \vdash \psi}{\Gamma \vdash \phi \vee \psi}$$

$$\frac{\Gamma \vdash A : t}{\Gamma \vdash \textit{Right } A : \textit{Either } t_0 \ t} \text{ (Right)}$$

## Сравним (4)

$$\text{(Disj E)} \frac{\Gamma \vdash \phi \vee \psi \quad \phi, \Gamma \vdash \nu \quad \psi, \Gamma \vdash \nu}{\Gamma \vdash \nu}$$

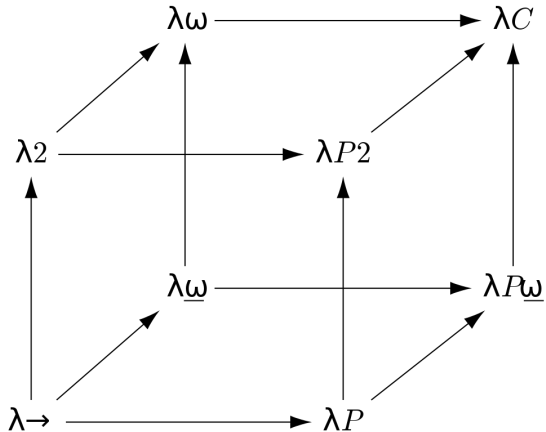
$$\frac{\Gamma \vdash A : \textit{Either } t_1 \ t_2 \quad x : t_1, \Gamma \vdash B : t_3 \quad y : t_2, \Gamma \vdash C : t_3}{\Gamma \vdash \underline{\textit{case}} \ A \ \underline{\textit{of}} \ \textit{Left } x \rightarrow B \ \underline{\textit{or}} \ \textit{Right } y \rightarrow C} \text{(Case)}$$



# Наблюдаемые соответствия

$\lambda$ -исчисление	Формальная логика
Типы	Теоремы
Выражения	Доказательства
Импликация	Конструктор типов функций
Конъюнкция	Тип кортежа
Дизъюнкция	Тип Either

- Наблюдаемая структурная эквивалентность между доказательствами и программами
- Можно формализовать в виде изоморфизма между логической системой и типизированным исчислением



$\lambda \rightarrow$	—	STLC
$\lambda 2$	—	STLC + полиморфизм
$\lambda \underline{\omega}$	—	STLC + алгебраические типы
$\lambda P$	—	STLC + зависимые типы

- Типовая система достаточно мощная для кодирования содержательных теорем
- Доказывать теоремы (синтезировать выражение заданного типа) можно в полуавтоматическом режиме
- Подобные языки: CoQ, Agda, Idris и т.д.

Рассмотрим три различных решения одной задачи взятия первого и последнего элементов списка

- Классический
- “Безопасный”
- С призрачными типами

## Решение #1

```
head :: [a] -> a
head xs = case xs of
  (x:_) -> x
  [] -> error "empty list!"

endpts = do
  putStrLn "Enter a non-empty list of integers:"
  xs <- readLn
  if xs /= [] then return (head xs, head (reverse xs))
  else endpts
```

## Решение #2

```
headMay :: [a] -> Maybe a
```

```
headMay xs = case xs of
```

```
  (x:_) -> Just x
```

```
  [] -> Nothing
```

```
safeEndpts = do
```

```
  putStrLn "Enter a non-empty list of integers:"
```

```
  xs <- readLn
```

```
  case headMay xs of
```

```
    Just x -> return (x, fromJust (headMay (reverse xs)))
```

```
    _ -> safeEndpts
```

## Решение #3

```
rev_cons :: Proof (IsCons xs) -> Proof (IsCons (Rev xs))  
gdpReverse :: ([a] ~~ xs) -> ([a] ~~ Rev xs)
```

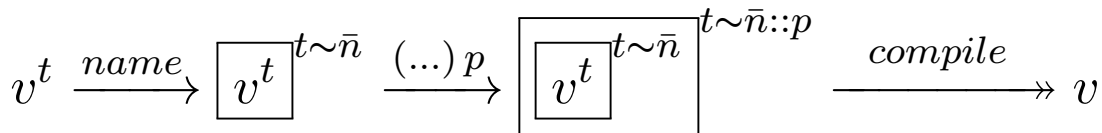
```
gdpHead :: ([a] ~~ xs ::: IsCons xs) -> a  
gdpHead xs = head (the xs) - safe!
```

```
gdpEndpts = do  
  putStrLn "Enter a non-empty list of integers:"  
  xs <- readLn  
  name xs $ \xs -> case classify xs of  
    IsCons proof ->  
      return (gdpHead (xs ...proof),  
              gdpHead (gdpReverse xs ...rev_cons proof))  
    IsNil proof -> gdpEndpts
```



- #1 — система типов не убережет от ошибки
- #2 — избыточные конструкторы
- #3 — использование лишних структур на уровне типов?

## Подробнее о призрачных типах



# Примитивы призрачных доказательств

```
data Proof p = QED
```

```
data TRUE
```

```
data FALSE
```

```
data p && q
```

```
data p || q
```

```
data p -> q
```

```
data Not p
```

```
data p == q
```

# Правила вывода призрачных доказательств

```
andIntro      :: Proof p -> Proof q -> Proof (p && q)
andElimL      :: Proof (p && q) -> Proof p
orIntroL      :: Proof p -> Proof (p || q)
implIntro     :: (Proof p -> Proof q) -> Proof (p -> q)
implElim      :: Proof (p -> q) -> Proof p -> Proof q
notIntro      :: (Proof p -> Proof FALSE) -> Proof (Not p)
contradicts   :: Proof p -> Proof (Not p) -> Proof FALSE
absurd        :: Proof FALSE -> Proof p
refl          :: Proof (x == x)
               - ... and many more
```

```
axiom :: Proof p
axiom = QED
```

## Пример

```
reverse :: ([a] ~~ xs) -> ([a] ~~ Rev xs)
reverse xs = defn (Prelude.reverse (the xs))
```

```
rev_length :: Proof (Length (Rev xs) == Length xs)
rev_rev    :: Proof (Rev (Rev xs) == xs)
rev_cons   :: Proof (IsCons xs) -> Proof (IsCons (Rev xs))
```

## Пример (2)


```
zipWith :: (a -> b -> c) -> ([a] ~~ xs ::: Length xs == n)  
        -> ([b] ~~ ys ::: Length ys == n) -> [c]  
zipWith f xs ys = Prelude.zipWith f (the xs) (the ys)
```

## Пример (3)


```
dot :: ([Double] ~~ vec1 ::: Length vec1 == n)
    -> ([Double] ~~ vec2 ::: Length vec2 == n) -> Double
dot vec1 vec2 = sum (zipWith (*) vec1 vec2)
```

```
dot_rev :: [Double] -> Double
dot_rev xs = name xs $ \vec ->
    dot (vec ...refl) (reverse vec ...rev_length)
```

 [A Tutorial Introduction to the Lambda Calculus](#)  
*Raúl Rojas*  
[PDF](#)

 [Курс математической логики и теории вычислимости](#)  
*Герасимов А.С.*  
[PDF](#)

 [Типы в языках программирования. 1й том.](#)  
*Бенджамин Пирс*

 [Software Foundations \(vol. 1, 2\)](#)  
*Benjamin C. Pierce*  
[Online](#)





Ghosts of Departed Proofs

*Matt Noonan*

PDF