### The $\lambda$ -Calculus

Kian Dianati

March 2, 2022



Alonzo Church

## What is the $\lambda$ -calculus?

A way to formally express computable functions. Kinda like a Turing machine.

# Unlike in a Turing machine, you can "express"

functions in the  $\lambda$ -calculus.

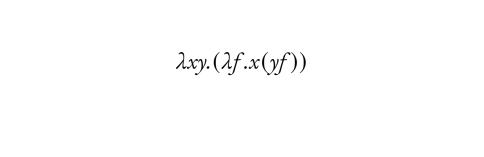
# So it's programming before programming

existed.

It is an alternative to Zermelo-Fraenkel Set Theory for the foundations of mathematics.

# How to $\lambda$ -calculus?

- Expressions
- ► Functions
- Applying expressions



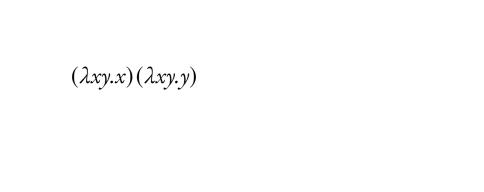
 $\lambda xy.(\lambda f.x(yf))$ 

decleration  $\rightarrow$ 

 $\lambda xy$ .

 $\lambda xy.(\lambda f.x(yf))$ 

decleration  $\rightarrow \lambda xy$ . expression  $\rightarrow \lambda f.x(yf)$ 



$$(\lambda xy.x)(\lambda xy.y) =_{\alpha} (\lambda xy.x)(\lambda pq.q)$$

 $(\lambda xy.x)(\lambda xy.y) =_{\alpha} (\lambda xy.x)(\lambda pq.q)$ 

 $\rightarrow_{\beta} (\lambda xy.x)[(\lambda pq.q)/x]$ 



$$(\lambda xy.x)(\lambda xy.y) =_{\alpha} (\lambda xy.x)(\lambda pq.q)$$

 $\rightarrow_{\beta} (\lambda xy.x)[(\lambda pq.q)/x]$ 

 $\rightarrow_{\beta} \lambda pq.q$ 

And we can also name expressions, for easier writing.

# Logic

 $\mathbf{t} = \lambda x y. x$  $\mathbf{f} = \lambda x y. y$ 

In **not**, we want the opposite of the input:

 $not = \lambda a.a$ 

In **not**, we want the opposite of the input:

 $not = \lambda a.af$ 

In **not**, we want the opposite of the input:

 $not = \lambda a.aft$ 

### and = $\lambda ab.ab\mathbf{f}$

### $\mathbf{and} = \lambda ab.ab\mathbf{f}$ $\mathbf{or} = \lambda ab.a\mathbf{t}b$

## $\mathbf{and} = \lambda ab.ab\mathbf{f}$ $\mathbf{or} = \lambda ab.a\mathbf{t}b$

**if** =  $\lambda cte.cte$ 

$$\mathbf{and} = \lambda ab.ab\mathbf{f}$$
$$\mathbf{or} = \lambda ab.a\mathbf{t}b$$

**if** =  $\lambda cte.cte$ 

 $\rightarrow_{\beta} \lambda x.x$ 

(and true false)

(define true (lambda (x y) x))
(define false (lambda (x y) y))

#cedure false>

(define and (lambda (a b) (a b false)))

# Numbers

In the  $\lambda$ -calculus we encode numbers using

higher order functions

$$\mathbf{o} = \lambda f x. x$$

$$\mathbf{o} = \lambda f x. x$$
$$\mathbf{i} = \lambda f x. f(x)$$

 $\mathbf{o} = \lambda f x. x$ 

 $\mathbf{I} = \lambda f x. f(x)$ 

 $\mathbf{z} = \lambda f x. f(f(x))$ 

 $\mathbf{o} = \lambda f x. x$ 

 $\mathbf{I} = \lambda f x. f(x)$ 

 $\mathbf{2} = \lambda f x. f(f(x))$ 

 $\mathbf{3} = \lambda f x. f(f(f(x)))$ 

 $\mathbf{o} = \lambda f x. x$ 

 $\mathbf{I} = \lambda f x. f(x)$ 

 $\mathbf{2} = \lambda f x. f(f(x))$ 

 $\mathbf{n} =_{\alpha} \lambda f x. f^{n}(x)$ 

 $\mathbf{3} = \lambda f x. f(f(f(x)))$ 

### To $m \times n$ :

### To $m \times n$ :

$$\lambda f x. f^n(x)$$

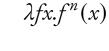
## To $m \times n$ :

$$\lambda f x. f^n(x)$$



### To $m \times n$ :

$$10 m \times n$$
.



 $\lambda fx.(f^m)^n(x)$ 

### nm

### $\mathbf{nm} =_{\alpha} (\lambda f x. f^{n}(x))(\lambda pq. p^{m}(q))$

$$\mathbf{nm} =_{\alpha} (\lambda f x. f^{n}(x))(\lambda pq. p^{m}(q))$$

 $\mathbf{nm} =_{\alpha} (\lambda f x. f^{n}(x))(\lambda p q. p^{m}(q))$ 

 $\rightarrow_{\beta} \lambda x.(\lambda pq.p^m(q))^n(x)$ 

 $\mathbf{nm} =_{\alpha} (\lambda f x. f^{n}(x))(\lambda p q. p^{m}(q))$ 

 $\rightarrow_{\beta} \lambda x.(\lambda pq.p^m(q))^n(x)$ 

 $\rightarrow_{\beta} \lambda xq.(x^m)^n(q)$ 

 $\mathbf{nm} =_{\alpha} (\lambda f x. f^{n}(x))(\lambda p q. p^{m}(q))$ 

 $\rightarrow_{\beta} \lambda x.(\lambda pq.p^{m}(q))^{n}(x)$ 

 $\rightarrow_{\beta} \lambda xq.(x^m)^n(q)$ 

 $\rightarrow_{\beta} \lambda xq.x^{nm}(q)$ 

# Two interesting things to think about.

### **RR**

$$\mathcal{L}(\mathcal{L}(\mathcal{L}(\mathcal{L}))) = \mathcal{L}(\mathcal{L}(\mathcal{L}(\mathcal{L})))$$

 $\mathbf{RR} \to_{\beta} \lambda x.\mathbf{not}(xx)[\mathbf{R}/\mathbf{x}]$ 

$$\operatorname{Re}(\lambda x.\operatorname{Hot}(xx) - \mathbf{K})$$

 $\mathbf{RR} \to_{\beta} \lambda x.\mathbf{not}(xx)[\mathbf{R}/\mathbf{x}]$ 

 $\rightarrow_{\beta} not(RR)$ 

$$\mathbf{Y} = \lambda f x. (\lambda x. f(xx)) (\lambda x. f(xx))$$

## And so $\mathbf{Y}g \to_{\beta} g(g(g(\ldots)))$

Could this mean transfinite cardinals in the λ-calculus?

# Thanks for listening!