

While we wait to get started

1 - Find someone to work with

2 - Clone from Github:

<https://github.com/bricef/Using-and-Abusing-Ruby>

Then run

bundle install --path vendor/bundle

BRICE FERNANDES @FRACTALLAMBDA

USING AND ABUSING RUBY FOR COMPUTER SCIENCE GREAT GOOD













→ X { ... } [...]

Proc



All The Things!



KEEP
CALM
IT'S
DEMO
TIME

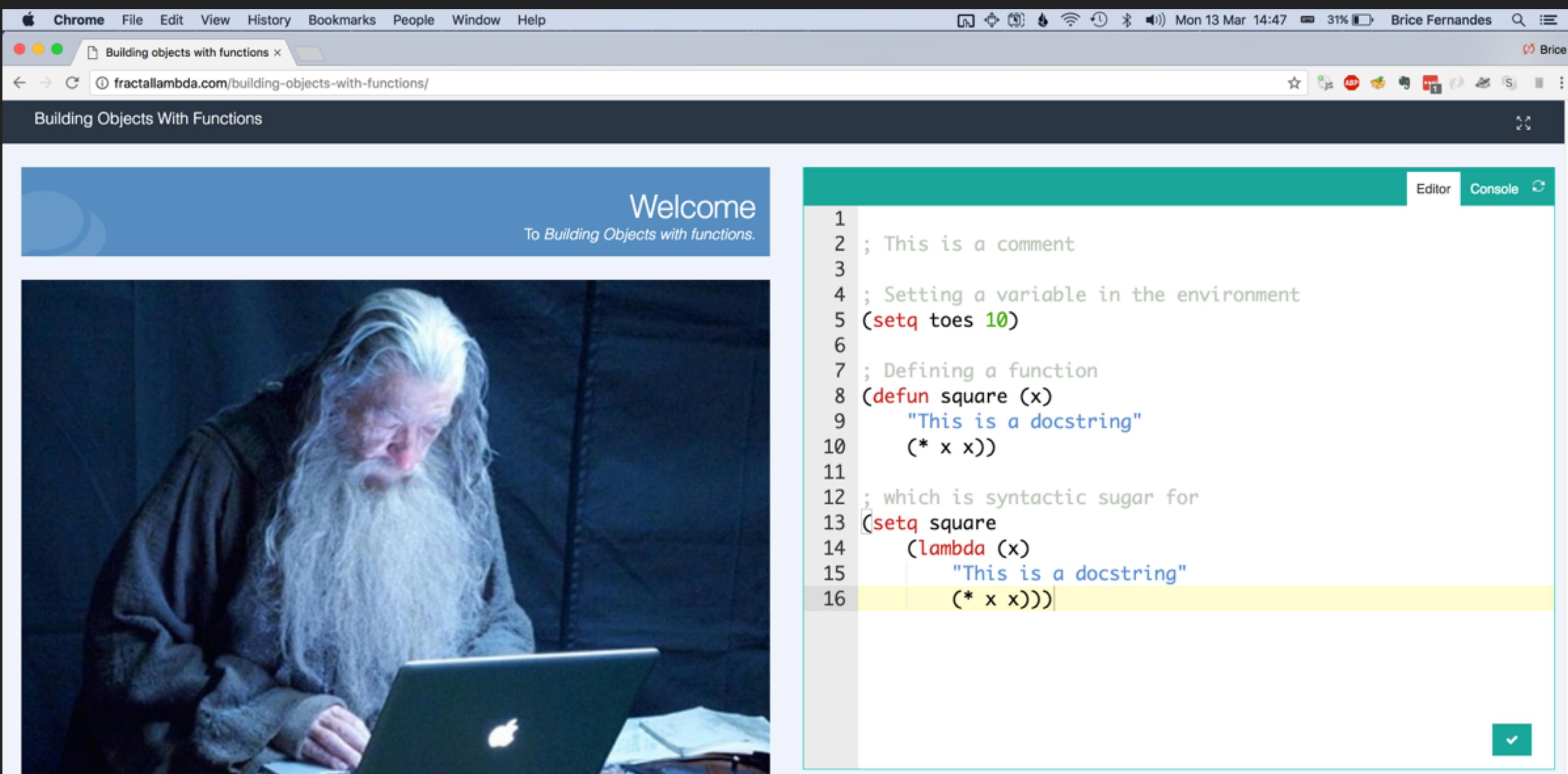
Procs
&
Currying



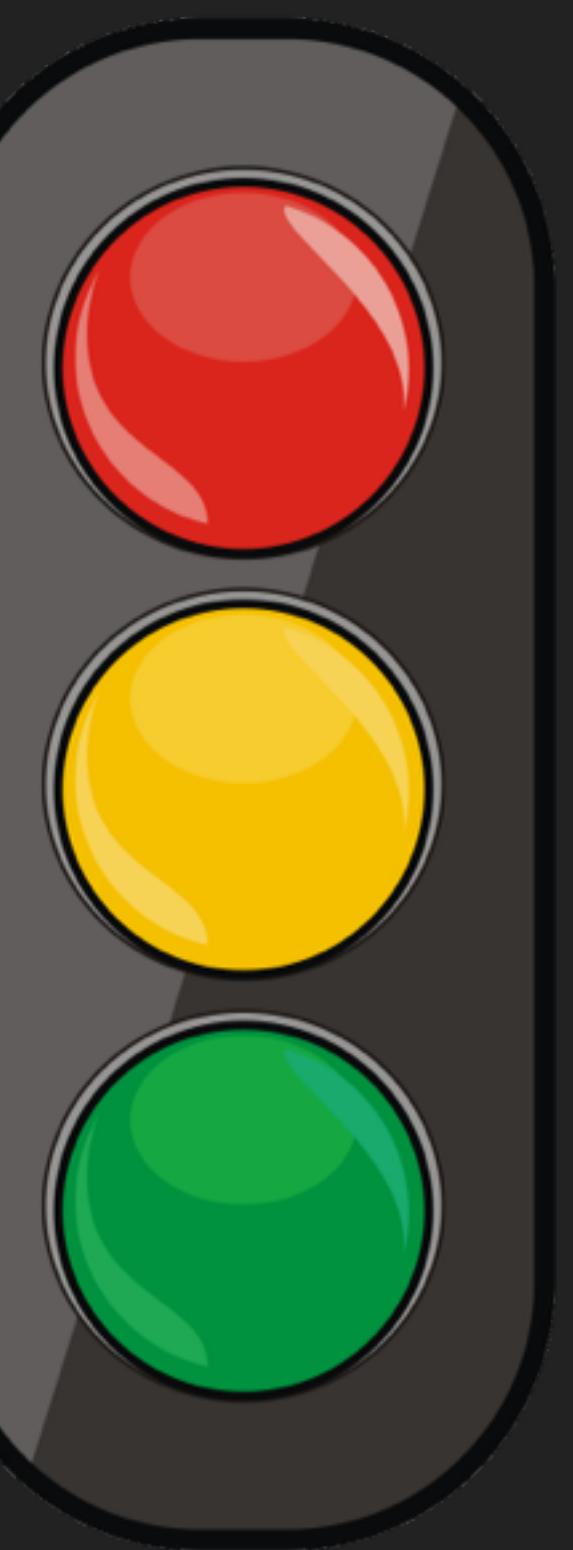
if Bool and
list equal not
first pair math
rest or Ints

We can build an object oriented language with these features!

<http://fractallambda.com/building-objects-with-functions/>



```
1 ; This is a comment
2
3 ; Setting a variable in the environment
4 (setq toes 10)
5
6 ; Defining a function
7 (defun square (x)
8   "This is a docstring"
9   (* x x))
10
11 ; which is syntactic sugar for
12 (setq square
13   (lambda (x)
14     "This is a docstring"
15     (* x x)))
```



Clone

<https://github.com/bricef/Using-and-Abusing-Ruby>

Then run

```
bundle install --path vendor/bundle
```

1

Truth & Logic

Proc TRUE, when given A and B returns A

Proc FALSE, when given A and B returns B

TRUE = $\lambda a \{ \lambda b \{ \dots \} \}$

FALSE = $\lambda a \{ \lambda b \{ \dots \} \}$

TRUE[1][2] # \rightarrow 1

FALSE[1][2] # \rightarrow 2



KEEP
CALM
IT'S
DEMO
TIME

True
&
False

if Bool and
list equal not
first pair math
rest or Ints

1

Truth & Logic

Proc TRUE, when given A and B returns A

Proc FALSE, when given A and B returns B

Proc AND, when given A and B ...

Proc OR, when given A and B ...

Proc NOT, when given A ...

Proc IF, when given C returns A if C else B

1

Truth & Logic

AND = $\rightarrow a \{ \rightarrow b \{ \dots \} \}$

OR = $\rightarrow a \{ \rightarrow b \{ \dots \} \}$

NOT = $\rightarrow a \{ \dots \}$

IF = $\rightarrow \text{condition} \{$

$\rightarrow \text{consequence} \{$

$\rightarrow \text{alternative} \{ \dots \} \}$



KEEP
CALM
IT'S
DEMO
TIME

If, And, Or
& Not

if

Bool

and

list

equal

not

first

pair

math

rest

or

Ints

2

Data Structures

Proc PAIR, LEFT and RIGHT

Proc PAIR takes two things and returns a pair

Proc LEFT takes a pair and returns the first item

Proc RIGHT takes a pair and ...

Hint: PAIR must return a Proc, and the Proc must
remember both items somehow

2

Data Structures

PAIR = → a { → b {...}}

LEFT = → pair {...}

RIGHT = → pair {...}

my_pair = PAIR[1][2]

LEFT[my_pair] #→ 1

RIGHT[my_pair] #→ 2



KEEP
CALM
IT'S
DEMO
TIME

Pair, Left
& Right

if

Bool

and

list

equal

not

first

pair

math

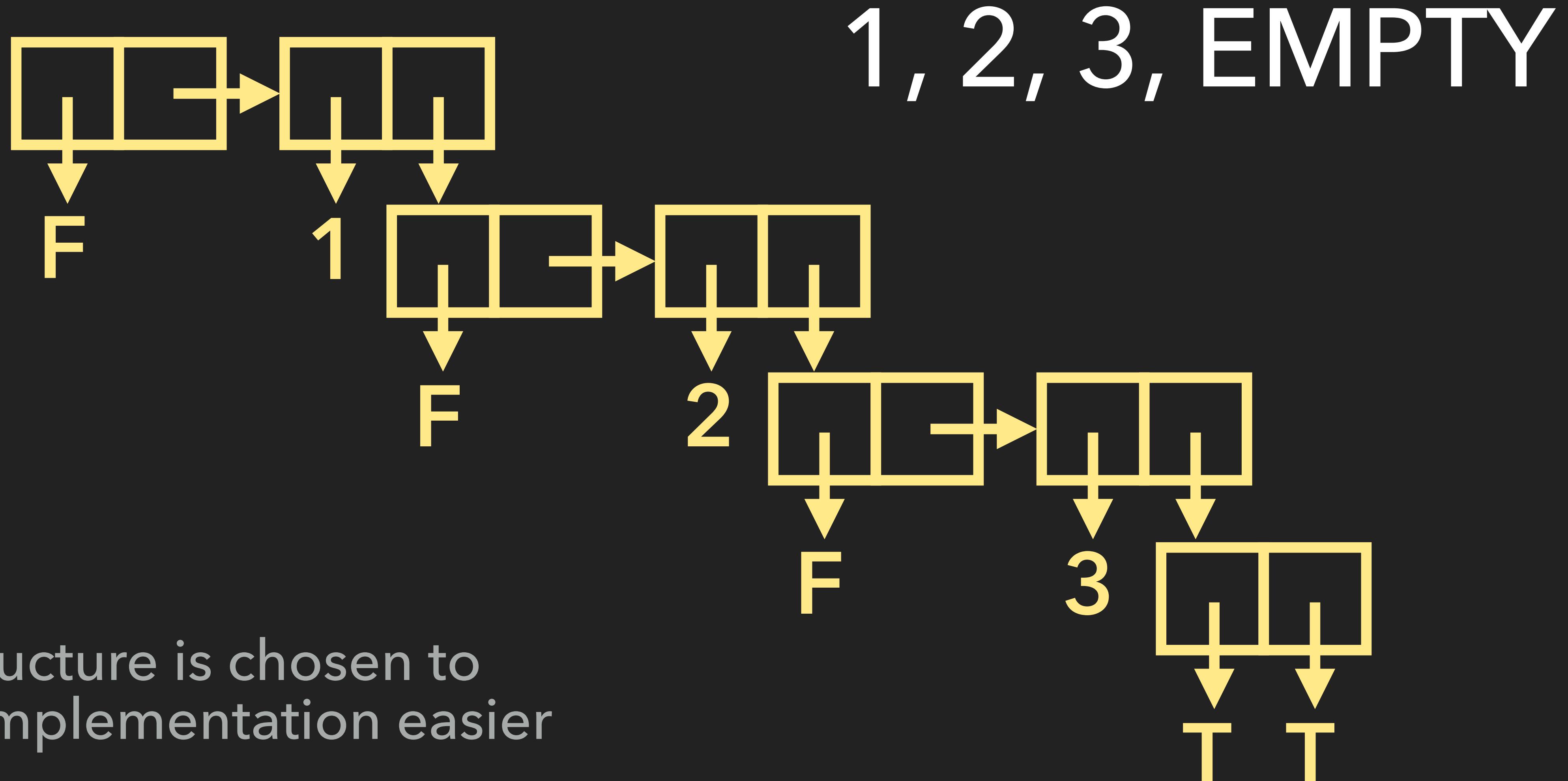
rest

or

Ints

2

Data Structures - List example



This structure is chosen to make implementation easier

2

Data Structures - List example

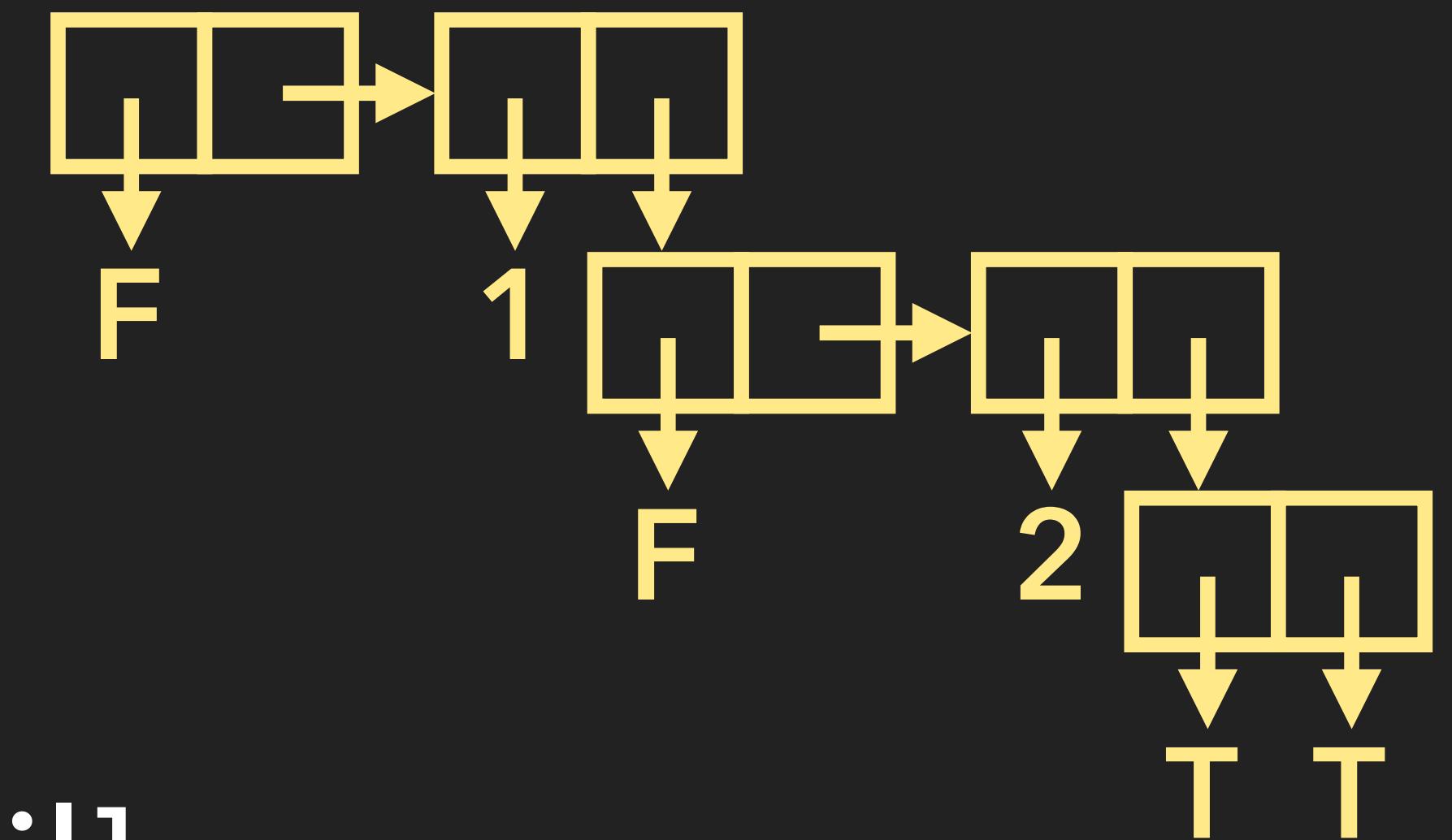
EMPTY is PAIR[TRUE][TRUE]

Lists nodes are nested PAIRS,

with the LEFT as FALSE

and the RIGHT as PAIR[head][tail]

CONS adds an item to the beginning of a list



Define the CONS, HEAD, TAIL and IS_EMPTY Procs



KEEP
CALM
IT'S
DEMO
TIME

List,
Empty?,
Head
& Tail

if Bool and
list equal not
first pair math
rest or Ints

3

Numbers

A number is a Proc, than when passed in a Proc and a value, applies the Proc n time to the value

$$\text{ZERO}(f, x) = x$$

$$\text{ONE}(f, x) = f(x)$$

$$\text{TWO}(f, x) = f(f(x))$$

3

Numbers

A number is a Proc, than when passed in a Proc and a value, applies the Proc n time to the value

ZERO(f ,x) = x	→ f {
ONE(f ,x) = f(x)	→ x {
TWO(f ,x) = f(f(x))	... }



KEEP
CALM
IT'S
DEMO
TIME

Church
Numerals

if Bool and
list equal not
first pair math
rest or Ints

3

Math

ADDONE is a Proc which will add one

ADD is a proc that will add two numbers

MINUSONE will return a subtract one

MINUS is a Proc that will subtract two numbers

MULT is a Proc that will multiply two numbers

EXP is a Proc that will raise a number to a power



KEEP
CALM
IT'S
DEMO
TIME

Math
operators

if Bool and
list equal not
first pair math
rest or Ints

4

Equal?

IS_ZERO is a Proc that returns TRUE if given ZERO

LEQ is a PROC that returns TRUE if $A \leq B$

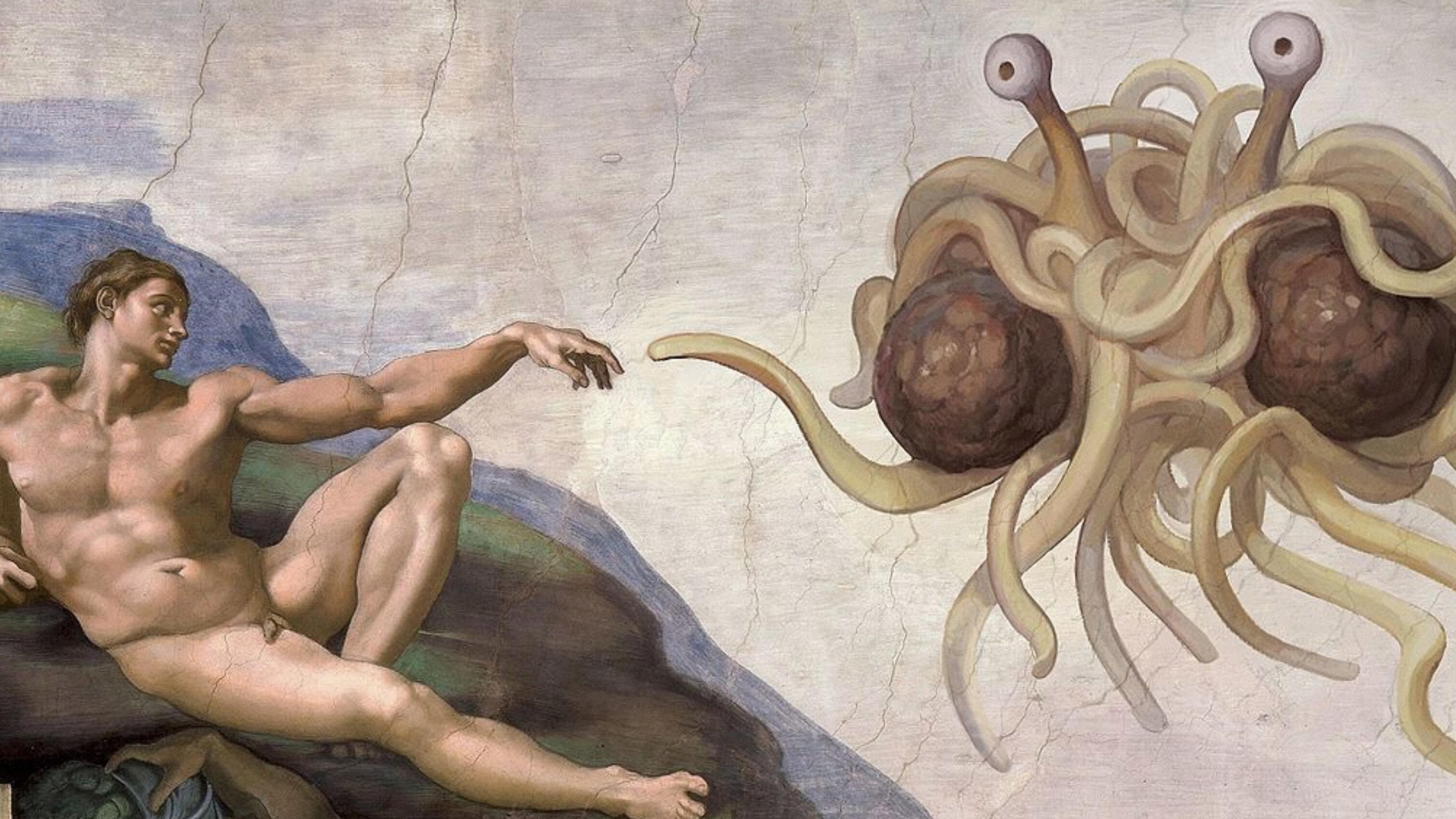
EQ is a PROC that returns TRUE if $A = B$



KEEP
CALM
IT'S
DEMO
TIME

Equal?

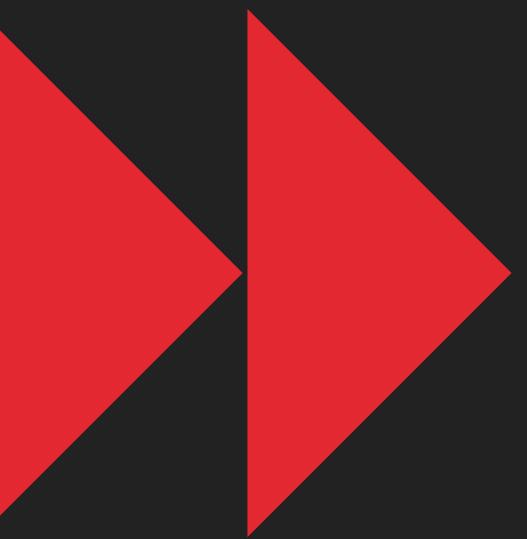
if Bool and
list equal not
first pair math
rest or Ints



IO?

Loops/Recursion?

Strings?



List operations?

(reduce)

Dude, Where's
my float?

IO?

Strings?

List operations?

(reduce)

Loops/Recursion?

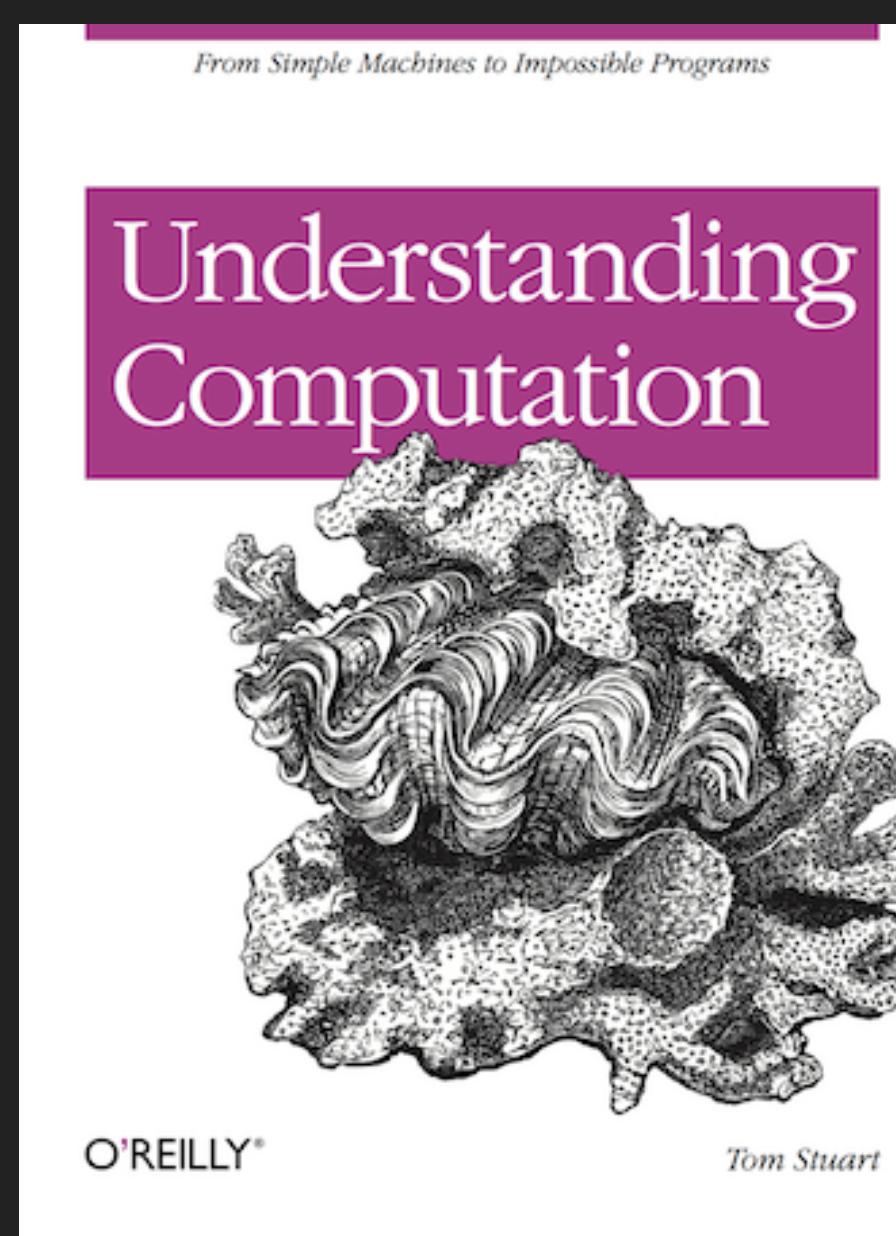


So wat?

Dude, Where's
my float?

Z?

Y?



Understanding
Computation
(Tom Stuart)

SICP
(Abelson &
Sussmans)

On Lisp
(Paul Graham)

<http://tinyurl.com/programming-with-nothing>
<http://codon.com/programming-with-nothing>
<http://tinyurl.com/objects-with-functions>
<http://fractallambda.com/building-objects-with-functions/>

THANK YOU!

BRICE FERNANDES

@FRACTALLAMBDA

BRICE@FRACTALLAMBDA.COM

(I SUCK AT  - KEEP TRYING)