Announcements

Bonus Lecture!

- Bob Harper talking about combinators in the lambda calculus
- This Friday from 11:20-12:30 in GHC 4125

Password

- constructive

Functional Programming
&
Constructive Logic

Hype for Types, Lecture 3

Logic

^	F	T
F	F	F
T	F	T

=>	F	T
F	T	T
T	F	T

A => B == -A v B

-(A v B) == (-A ^ -B)

...

v	F	T
F	F	T
T	T	T

⊕	F	T
F	F	T
T	T	F

- Nothing new
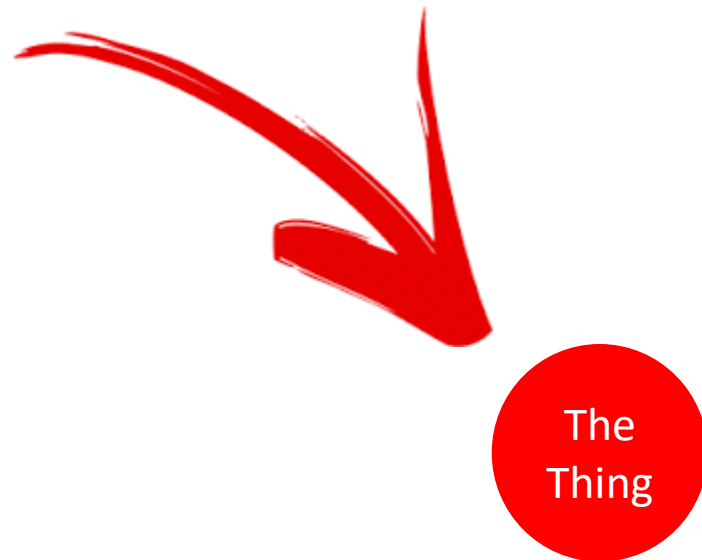- 151/127/128 flashbacks

What if we made logic

* ***better*** *

Take logic, and make one sensible addition

A sensible statement

"If you wish to prove to me that something exists, you must show me the thing."

The Thing

Examples

- Do unicorns exist?
  - Proof 1: Here is a unicorn:

  - Proof 2: Assume unicorns did not exist. Then, no magical animal would ride on rainbows. But we know there is a magical animal that rides on rainbows. Contradiction! Therefore, unicorns exist.

Examples

- Did you do your homework?
  - Proof 1: Here is my homework submission:

    1 - jluningp@andrew.cmu.edu_1_handin.sml

  - Proof 2: Assume you did not do your homework. Then we would be sad. But we are not sad. Contradiction! Therefore, you did your homework.

Examples

- Can you write code that parses lambda calculus?
  - Proof 1: Here is a parser

```
sml
name LambdaCalculusParseFun

terminal IDENT of string

terminal LPAREN
terminal RPAREN

terminal LAMBDA
terminal DARROW

nonterminal Exp : exp =
  1:ExpOne => id_exp
  1:Exp 2:ExpOne => Apply

nonterminal ExpOne : exp =
  1:IDENT => Variable
  LPAREN 1:Exp RPAREN => id_exp
  LAMBDA 1:IDENT DARROW 2:Exp => Lambda

start Exp
```

  - Proof 2: Assume you can't. [A couple properties of lambda calculus later]. Contradiction! Therefore, you can parse lambda calculus.

The difference between Proof 1 and Proof 2 is information content.

- Do unicorns exist?
  - Proof 1: I have a *unicorn* now. And I know that it believes in itself!
  - Proof 2: uh…
- Did you do your homework?
  - Proof 1: I can now find and grade your homework.
  - Proof 2: uh…
- Can you write code that parses lambda calculus?
  - Proof 1: If I install cmyacc, I can parse lambda calculus now.
  - Proof 2: uh…

Another statement

"It is *not* enough to prove that *bad things* will happen if the thing did not exist."
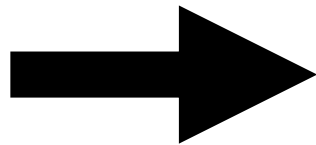
Constructive Logic

- The logic of "If you wish to prove that something exists, you must produce the thing."

- The logic of type theory, programming language theory, and computer science in general.

- Our namesake: 15-317

Now for some more rules

Why constructive?

- In programming, we must actually compute and construct things.

- When I prove something using code, that code actually computes the proof
  - Example:

$$(\lambda(x : A)\ M) : A \rightarrow B$$

is a function from a proof of A to a proof of B. If I call this function, it actually has to produce the proof of B.