Did I get it right?

COS 326
David Walker
Princeton University

http://~cos326/notes/evaluation.php
http://~cos326/notes/reasoning.php

slides copyright 2026 David Walker
permission granted to reuse these slides for non-commercial educational purposes

Last Time: Informal Proofs for e1 = e2

Key Idea: Definition of equality

- e1 = e2 iff e1 and e2 evaluate to the same value

Proof Rules for Equality:

- evaluation: if e1 -> e2 then e1 = e2
  - reflexivity, symmetry, and transitivity
- other common facts about functions (eg: 1*x = x)
- substitution of equals for equals: e1 = e2 implies foo(e1) = foo(e2)

Proof Rule for Universal Quantifiers (all x:t. F)

- consider arbitrary x:t and prove F

Example:

Theorem: for all integers n and m, easy 1 n m == n + m
Proof: Consider arbitrary n, m:

	easy 1 n m	(LHS)
=	1 * (n + m)	(by evaluating easy)
=	n + m	(by math)

Last Time: Lean Intro

In Lean is a functional programming language:

Lean:

```
inductive B : Type where
| true
| false
```

```
def not (b : B) : B :=
  match b with
  | B.true => B.false
  | B.false => B.true
```

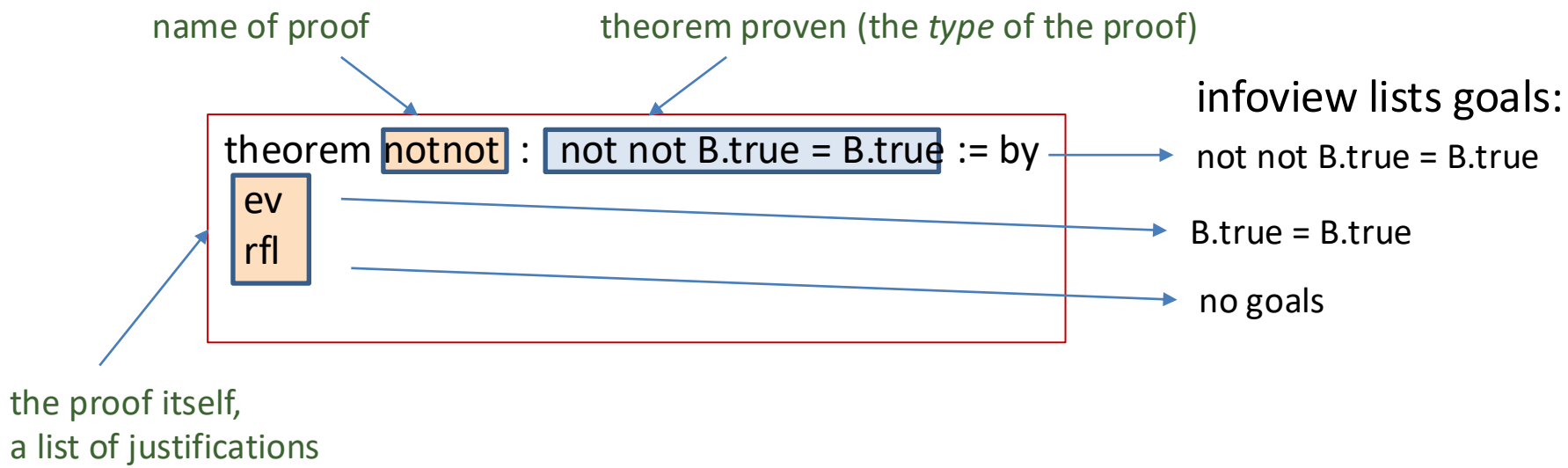
OCaml:

```
type b =
| True
| False
```

```
let not (b : b) : b :=
  match b with
  | True -> False
  | False -> True
```

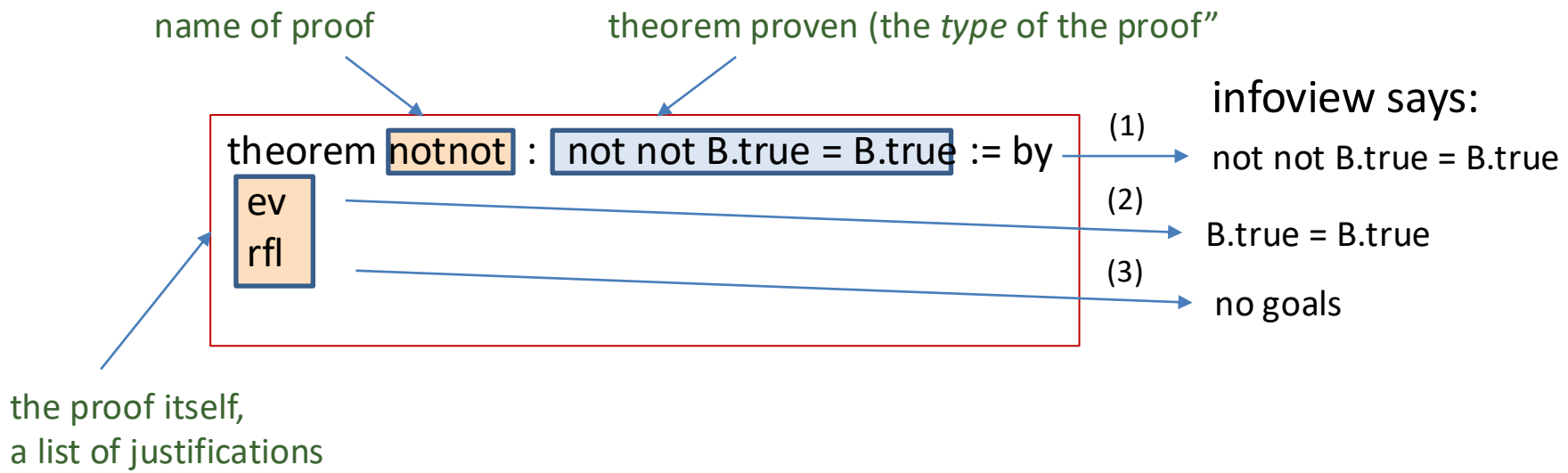
Last Time: From Informal Proofs to Formal Ones

Lean is also a theorem proving environment:



From Informal Proofs to Formal Ones

Lean is also a theorem proving environment:



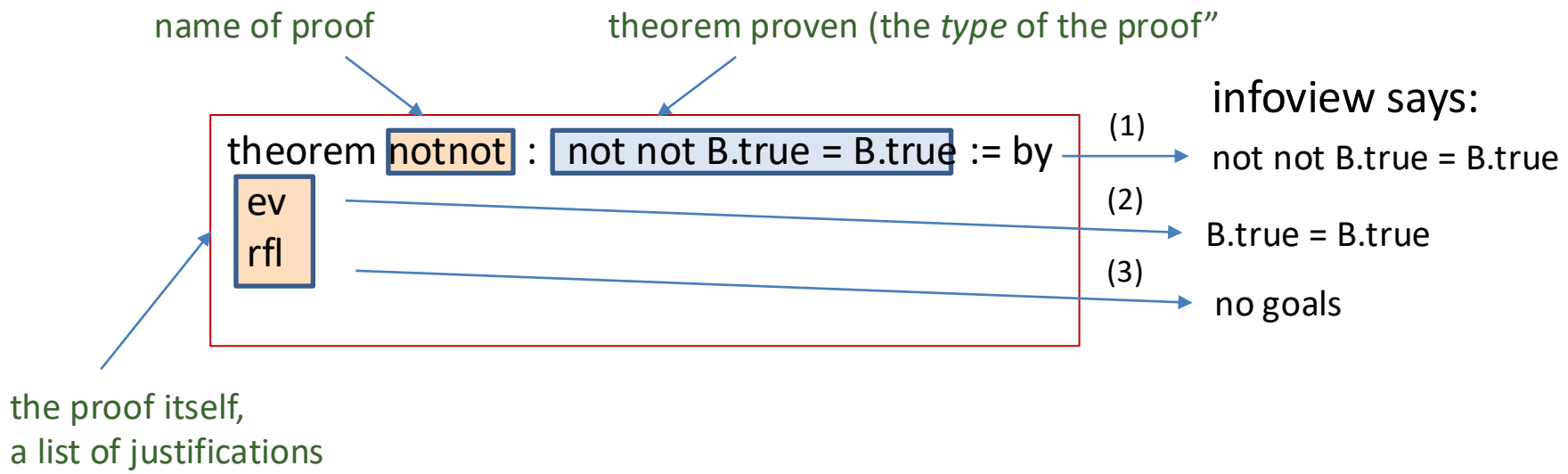
informal rendering of that proof:

theorem not not B.true = B.true:

(1) not not B.true = B.true	Must prove this! let's go!
(2) B.true = B.true	By ev, it suffices to prove this
(3) QED	By reflexivity, we are done

From Informal Proofs to Formal Ones

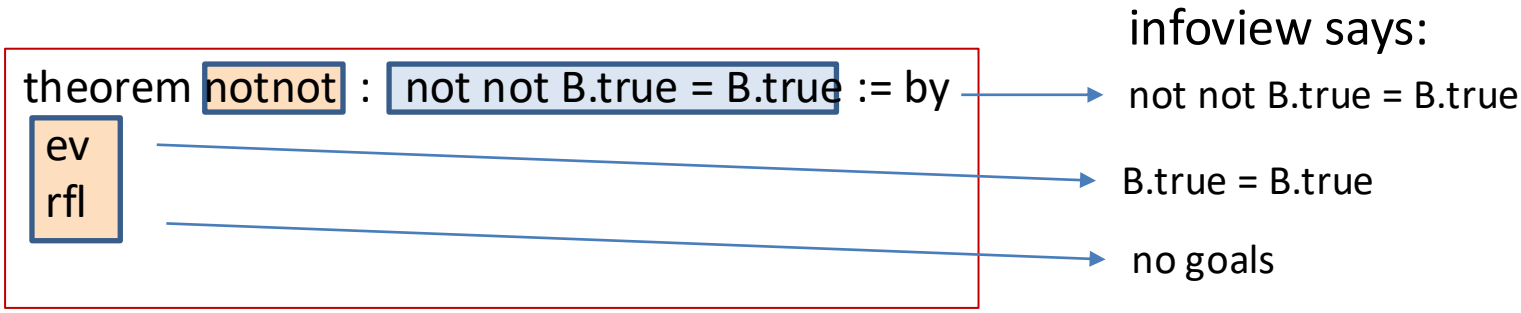
Lean is also a theorem proving environment:



another informal rendering of that proof:

```
theorem not not B.true = B.true:
    not not B.true          (LHS)
  = B.true                  by ev(aluation)
  = B.true                  by rfl (reflexivity) this is the RHS
```

From Informal Proofs to Formal Ones



Overall Picture

Informal proofs

- Write down properties, each one with a justification
- Use shorthand, conventions, English and common knowledge justifications

Lean proofs

- Write down **justifications**; Infoview calculates what is left to be proven

```
theorem not not B.true = B.true:
    not not B.true
  = B.true                  (ev)
  = B.true                  (rfl)
```