

Definition

The rule of *referential transparency*:

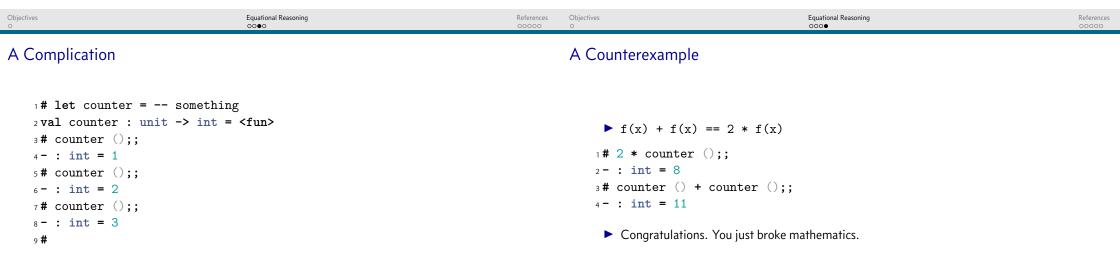
$$\frac{e_1 \rightarrow^* v \quad e_2 \rightarrow^* v \quad f e_1 \rightarrow^* w}{f e_2 \rightarrow^* w}$$

If you have two expressions that evaluate to be the same thing then you can use one for the other without changing the meaning of the whole program.

 You can prove this by induction, using the natural semantic rules from the previous lectures. > You can use equational reasoning to make the following equivalence:

 $f(if e_1 then e_2 else e_3) \equiv if e_1 then f(e_2) else f(e_3)$

- $1 \mathbf{x} *$ (if foo then 20 / x else 23 / x) -- equivalent to 2 if foo then 20 else 23 -- well, mostly
- > You have the basis now of many compiler optimization opportunities!



• Can we still use equational reasoning to talk about programs now?

	< 口 > < 合 > < 분 > < 분 >	≡ ∽へぐ			<□><0><0><0><0><0><0><0><0><0><0><0><0><0>
Objectives o	Equational Reasoning 0000	References •0000	Objectives O	Equational Reasoning 0000	References ⊙●○○○
Reference Operator			Natural Semantics		
Transition Semantics ref $v \rightarrow \$i$, where $\$i$ is a free location ! $\$i \rightarrow v$, if state location $\$i$ contains v $\$i := v \rightarrow ()$, and state location $\$i$ is i (); $e \rightarrow e$ Note that references are different than p assigned to and read from.	<i>.</i>		$\frac{e \Downarrow v}{\text{ref } e \Downarrow \$i}, \text{ where }\$i \text{ is a free location}$ $\frac{e \Downarrow \$i}{!e \Downarrow v}, \text{ if state location }\$i \text{ contains}$ $\frac{e_1 \Downarrow \$i e_2 \Downarrow v}{e_1 := e_2 \Downarrow ()}, \text{ and location }\$i \text{ is set}$ $\frac{e_1 \Downarrow () e_2 \Downarrow v}{e_1; e_2 \Downarrow v}$	v.	

Objectives o	Equational Reasoning 0000	References Obje OO●OO O	ectives	Equational Reasoning 0000	References 000●0
Counter, Method 1		Ba	ad Things for Counter		
<pre>1# let ct = ref 0;; 2val ct : int ref = {conten 3# let counter () = 4 ct := !ct + 1; 5 !ct;; 6val counter : unit -> int 7# counter ();; 8- : int = 1 9# counter ();; 10- : int = 2</pre>			 ct is globally defined. Two bad things c 1. What if you already had a global va Correct solution: use modules. 2. The Stupid UserTM might decide ta Now your counter won't work lili Now you can't change the repre Remember the idea of abstraction 	ariable ct defined? o change ct just for fun. ke it's supposed to! sentation without getting tech support calls.	
	< □ > < A	・ 4 臣 > 4 臣 > 臣 - 夕久の		()) () () () () () () () () (E> E •94.0

References 0000●

Concl	usions	about	State
COLICI	usions	abbui	Slale

Objectives

State is bad because:

- ► It breaks our ability to use equational reasoning.
- Users can get to our global variables and change them without permission.

Equational Reasoning

State is good because:

- Certain constructs are almost impossible without state (e.g., graphs).
- Our world is a stateful one.