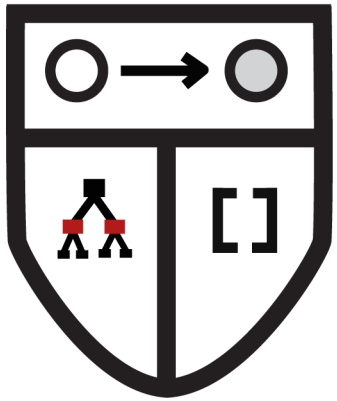


CS 3110

The Substitution Model

Prof. Clarkson

Fall 2019

Today's music: Substitute by The Who

CLICKER QUESTION 1

Review

Previously in 3110:

- interpreter for tiny calculator language
- lexing into tokens
- parsing into abstract syntax tree (AST)
- evaluation based on single steps

Today:

- finish evaluation
- extend to a bigger language: let expressions
- substitution model: a way to think about evaluation of `let`

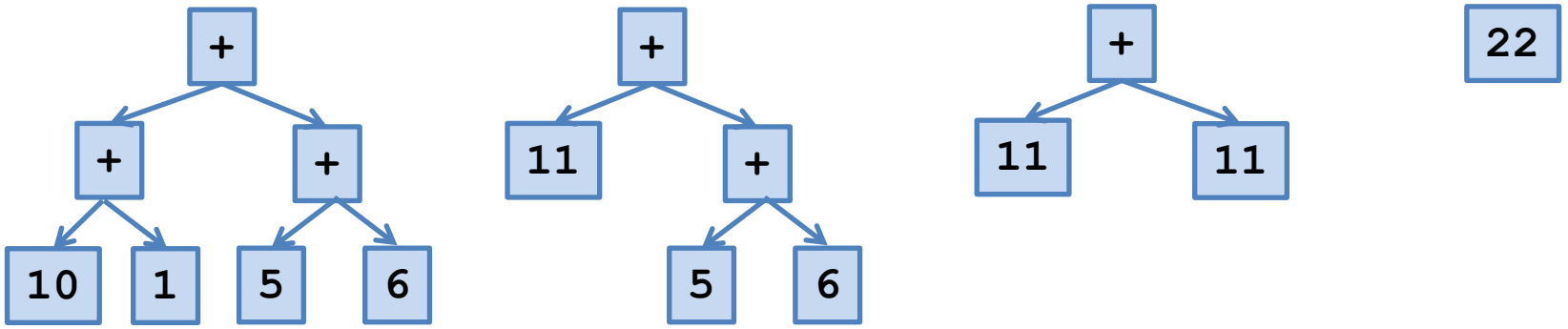
Review: calculator language BNF

```
e ::= i
    | e1 bop e2
    | ( e )

bop ::= + | *

i ::= integers
```

Demo

Review: evaluation strategy

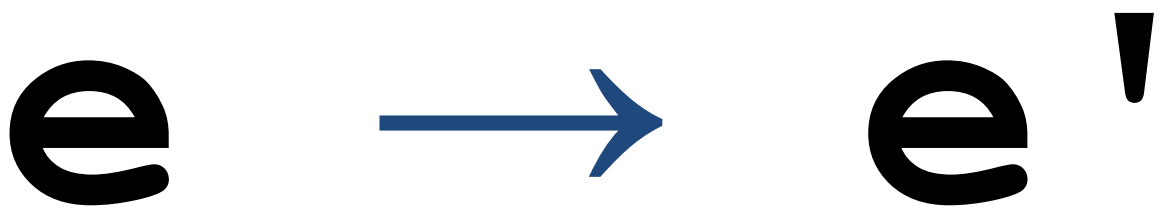
- An expression e takes a single *step* to a new expression e' by simplifying some subexpression
- Expression keeps stepping until it reaches a *value*
- Values never step further



Demo

CLICKER QUESTION 2

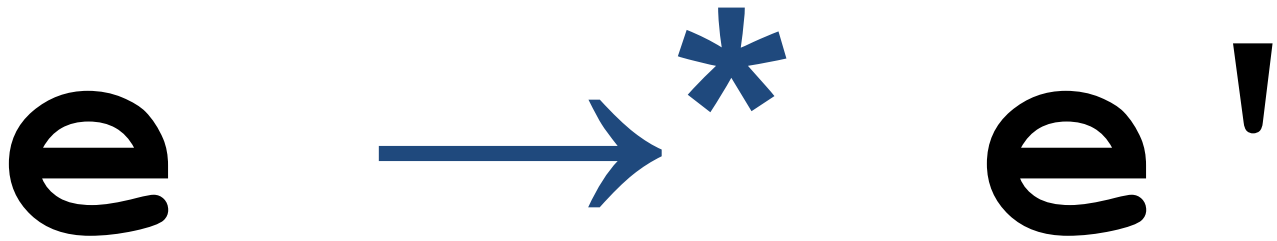
FORMAL DYNAMIC SEMANTICS



single-step relation

the `step` function we implemented

values never step



multi-step relation

related to the `eval` function we implemented

CLICKER QUESTION 3

Inductively defined relation

`e1 + e2` → `e1' + e2`

if **`e1` → `e1'`**

`v1 + e2` → `v1 + e2'`

if **`e2` → `e2'`**

`v1 + v2` → `i`

if **`i`** *is the result of primitive operation* **`v1 + v2`**

LET EXPRESSIONS

Let expressions

```
e ::= i
    | e1 bop e2
    | ( e )
    | x
    | let x = e1 in e2

bop ::= + | *
i ::= integers
x ::= identifiers
```

Let semantics

`let x = e1 in e2` → `let x = e1' in e2`

if `e1` → `e1'`

`let x = v1 in e2` →

`e2 with v1 substituted for x`

$$e\{v/x\}$$

means e with v substituted for x

Let semantics

$$\texttt{let x = e1 in e2} \rightarrow \texttt{let x = e1' in e2}$$

if $\texttt{e1} \rightarrow \texttt{e1'}$

$$\texttt{let x = v1 in e2} \rightarrow \texttt{e2\{v1/x\}}$$

Demo

Defining substitution: the easy parts

Nothing to do for integers:

i {v/x} = i

Just keep going through operations:

(e1 + e2) {v/x} = (e1 {v/x}) + (e2 {v/x})

Variables are where substitution really happens:

x {v/x} = v

y {v/x} = y

Defining substitution: let

$$(\text{let } y = e1 \text{ in } e2)\ \{v/x\}$$

$$=$$

$$\text{let } y = (e1\ \{v/x\}) \text{ in } (e2\ \{v/x\})$$

Do substitute in binding.
e.g.,
let x = 1 in
(let y = x in y)

$$(\text{let } x = e1 \text{ in } e2)\ \{v/x\}$$

$$=$$

$$\text{let } x = (e1\ \{v/x\}) \text{ in } e2$$

Stop substituting in body.
e.g.,
let x = 1 in
(let x = 2 in x)

gets even trickier in the presence of functions: see textbook re. capture-avoiding substitution

If expressions

```
e ::= x | i | b
    | e1 bop e2
    | let x = e1 in e2
    | if e1 then e2 else e3

bop ::= + | * | <=
```

If semantics

$$\texttt{if e1 then e2 else e3} \longrightarrow \texttt{if e1' then e2 else e3}$$
$$\texttt{if e1} \longrightarrow \texttt{e1'}$$

$$\texttt{if true then e2 else e3} \longrightarrow \texttt{e2}$$

$$\texttt{if false then e2 else e3} \longrightarrow \texttt{e3}$$

Demo

Upcoming events

- [last night] A5 due

This is not a substitute.

THIS IS 3110