6-1 J_2 — functions like Pascal or C or Java

CK_1 — global mapping of fun names to their defs

$\langle v_n, \text{kapp } (v_0 \ldots f)\ ()\ k \rangle$

$\mapsto \langle e_{body}', k \rangle$

where $e_{body}' = \text{subst } (x_0 \ldots x_n)\ (v_0 \ldots v_n)\ e_{body}$

define $f(x_0 \ldots x_n)\ e_{body} = \Delta(f)$

$\langle X, k \rangle \mapsto \ldots$ — variables are removed by subst code   kcontinuation

G-2/ CEK

env

$$st = \langle e, env, k\rangle$$
$$env = \emptyset \mid env[x \mapsto v]$$
$$k = kret \mid kif\ e\ e\ k$$
$$\mid kapp\ \vec{v}\ \vec{e}\ k$$

$$\langle x, env, k\rangle \mapsto \langle env(x), env, k\rangle$$
$$\langle if\ e_c\ e_t\ e_f, env, k\rangle \mapsto \langle e_c, env, kif\ e_t\ e_f\ k\rangle$$
$$\langle false, env, kif\ e_t\ e_f\ k\rangle \mapsto \langle e_f, env, k\rangle$$
$$\langle v, env, kif\ e_t\ e_f\ k\rangle \mapsto \langle e_t, env, k\rangle$$
$$\langle e_0\ e_m \ldots, env, k\rangle \mapsto \langle e_0, kapp\ ()\ (e_m \ldots)\ k\rangle$$
$$\langle v_1, env, kapp\ (v_0 \ldots)\ (e_0\ e_m \ldots)\ k\rangle$$
$$\mapsto \langle e_0, env, kapp\ (v_0 \ldots v_1)\ (e_m \ldots)\ k\rangle$$
$$\langle v_n, env, kapp\ (p\ v_0 \ldots)\ ()\ k\rangle$$
$$\mapsto \langle \delta(p, v_0 \ldots, v_n), env, k\rangle$$
$$\langle v_n, env, kapp\ (f\ v_0 \ldots)\ ()\ k\rangle$$
$$\mapsto \langle e_{body}, \overset{mt = \emptyset}{\cancel{env}}[x_0 \mapsto v_0] \ldots [x_n \mapsto v_n], k\rangle$$
where define $f(x_0 \ldots x_n)\ e_{body} \in \Delta(f)$

WRONG!!!

~~unless mt~~

6-3/ (define (Double x) (+ x x))

(Double 1)

$\langle \text{Double } 1,\ \emptyset,\ \text{kret} \rangle$

$\langle \text{Double},\ \emptyset,\ \text{kapp } ()\ (1)\ \text{kret} \rangle$

$\langle 1,\ \emptyset,\ \text{kapp } (\text{Double})\ ()\ \text{kret} \rangle$

$\langle (+\ x\ x),\ \emptyset[x \mapsto 1],\ \text{kret} \rangle$

$\langle +,\ \emptyset[x \mapsto 1],\ \text{kapp } ()\ (x\ x)\ \text{kret} \rangle$

$\langle x,\ \emptyset[x \mapsto 1],\ \text{kapp } (+)\ (x)\ \text{kret} \rangle$

$\langle 1,\ \emptyset[x \mapsto 1],\ \text{kapp } (+)\ (x)\ \text{kret} \rangle$

$\langle x,\ \emptyset[x \mapsto 1],\ \text{kapp } (+\ 1)\ ()\ \text{kret} \rangle$

$\langle 1,\ \emptyset[x \mapsto 1],\ \text{kapp } (+\ 1)\ ()\ \text{kret} \rangle$

$\langle 2,\ \emptyset[x \mapsto 1],\ \text{kret} \rangle \longrightarrow 2$

G-4/

```
(define (F x) y)
(define (G y) (F 0))
(G 1)
```

$\langle G\ 1,\ \emptyset,\ \text{kret} \rangle$

$\langle 1,\ \emptyset,\ \text{kapp}\ (G)\ ()\ \text{kret} \rangle$

$\langle F\ 0,\ \emptyset[y \mapsto 1],\ \text{kret} \rangle$

$\langle 0,\ \emptyset[y \mapsto 1],\ \text{kapp}\ (F)\ ()\ \text{kret} \rangle$

$\langle y,\ \emptyset[y \mapsto 1][x \mapsto 0],\ \text{kret} \rangle$

$\langle 1,$ " $,\ \text{kret} \rangle \longrightarrow 1$

$\langle y,\ \emptyset[x \mapsto 0],\ \text{kret} \rangle$

$\longrightarrow$ error!

JS — "this"

6-5/ emacs lisp — dynamic scope

```
(define (F x) true)
(if (F 0) x x)
```
$\Longrightarrow$ error!

$\langle$ if (F 0) x x , $\emptyset$, kret $\rangle$

$\langle$ F 0 , $\emptyset$, kif x x kret $\rangle$

$\langle$ 0 , $\emptyset$, kapp (F) () (kif x x kret) $\rangle$

$\langle$ true , $\emptyset[x \mapsto 0]$, kif x x kret $\rangle$

$\langle$ x , $\emptyset[x \mapsto 0]$, kret $\rangle$

$\langle$ 0 , " , kret $\rangle$ $\Longrightarrow$ 0

G-6/ correct CEK

$st = \langle e, env, k \rangle \qquad env = \emptyset \mid env[x \mapsto v]$

$k = kret$

$\quad \mid kif \; env \; e \; e \; k$

$\quad \mid kapp \; \vec{v} \; env \; \vec{e} \; k$

$\langle x, env, k \rangle \mapsto \langle env(x), \emptyset, k \rangle$

$\langle if \; ec \; et \; ef, env, k \rangle \mapsto \langle ec, env, kif \; env \; et \; ef \; k \rangle$

$\langle false, \_, kif \; env' \; et \; ef \; k \rangle \mapsto \langle ef, env', k \rangle$

$\langle v, \_, kif \; env' \; et \; ef \; k \rangle \mapsto \langle et, env', k \rangle$

$\langle e_0 \; em \ldots, env, k \rangle \mapsto \langle e_0, env, kapp \; () \; env \; (em \ldots) \; k \rangle$

$\langle v_1, \_, kapp \; (v_0 \ldots) \; env' \; (e_0 \; em \ldots) \; k \rangle$

$\quad \mapsto \langle e_0, env', kapp \; (v_0 \ldots v_1) \; env' \; (em \ldots) \; k \rangle$

$\langle vn, \_, kapp \; (p \; v_0 \ldots) \; \_ \; () \; k \rangle$

$\quad \mapsto \langle \delta(p, v_0 \ldots vn), \emptyset, k \rangle$

$\langle vn, \_, kapp \; (f \; v_0 \ldots) \; \_ \; () \; k \rangle$

$\quad \mapsto \langle eb, \emptyset[x_0 \mapsto v_0] \ldots [x_n \mapsto vn], k \rangle$

$\quad$ where $\Delta(f) = define \; f(x_0 \ldots x_n) \; eb$

G-7/ $J_2 \Rightarrow J_3$

move beyond C/Pascal
to functions like JS

"lambda functions"

↳ anonymous functions
that are locally scoped

JS : `(x) => 1 + x`

Py : `lambda : x : 1 + x`

C++ : `[](int x) { return 1+x; }`

6-8/ J_3: $e = v \mid e\,e \ldots \mid \text{if } e\,e\,e \mid x$

$v = b \mid (\lambda\ (x \ldots)\ e)$

$b = \text{num} \mid \text{bools} \mid \text{prim}$

$E = \text{hole} \mid \text{if } E\ e\ e \mid v \ldots E\ e \ldots$

$$E[(\lambda\ (x_0 \ldots x_n)\ e_b)\ v_0 \ldots v_n)] =$$
$$E[e_b[x_0 \leftarrow v_0] \ldots [x_n \leftarrow v_n]]$$

$((\lambda (x)$
$((\lambda (y)$
$(x + y))\ 7))\ 8)$

$\rightsquigarrow 8 + 7 = 15$

let $x = e_1$ in e_2

$\Rightarrow$

$(\lambda x . e_1)\ e_2$

$((\lambda\ (x)\ e_1)\ e_2)$

6-9)

let $x = 8$ in
let $y = 7$ in
$x + y$

let $x = 8$ in
let $x = x + 1$ in
$x + x$

let $[x_0\ e_0] \ldots [x_n\ e_n]$ in e_b
$\Rightarrow$
$((\lambda\ (x_0 \ldots x_n)\ e_b)\ e_0 \ldots e_n)$

let* in e_b $\Rightarrow$ e_b
let* $[x_0\ e_0]\ [x_m\ e_m] \ldots$ in e_b $\Rightarrow$
let $[x_0\ e_0]$ in let* $[x_n\ e_m] \ldots$ in e_b

6-10/

let $f =$
let $x = 1$ in
$\lambda y.\ x + y$
in
$f\ 3$

$=$

let $f =$
$\lambda y.\ 1 + y$
in
$f\ 3$
$=$
$1 + 3 = 4$

$CEK_0 : \quad v = b$

6-11/ $J_3 : v = b \mid \lambda(x \ldots)\, e$

$CEK_1 \quad v ::= b \mid clo(\lambda(x \ldots)\, e, env)$

$\langle \lambda(x \ldots)\, e, env, k \rangle$

$\quad \mapsto \langle clo(\lambda(x \ldots)\, e, env), \emptyset, k \rangle$

$\langle v_n, -, kapp((clo(\lambda(x_0, \ldots, x_n)\, e_b, env'),$
$\qquad v_0, \ldots) - () \quad k) \rangle$

$\quad \mapsto \langle e_b, env'[x_0 \mapsto v_0] \ldots [x_n \mapsto v_n], k \rangle$

6-12) ${}^{A}(\text{let } f = {}^{B}(\text{let } x = 1 \text{ in } {}^{C}(\lambda y. {}^{D}(x+y))) \text{ in}$

${}^{E}(f\ 3))$

$\langle A, \emptyset, \text{kret} \rangle$

$\langle B, \emptyset, \text{kapp } (\text{clo}(\lambda f. E, \emptyset))\ \_\ ()\ \text{kret} \rangle$

$\langle C, \emptyset[x \mapsto 1],$ " $\rangle$

$\langle \text{clo}(C, \emptyset[x \mapsto 1]), \_,$ " $\rangle$

$\langle E, \emptyset[f \mapsto \text{clo}(C, \emptyset[x \mapsto 1])], \text{kret} \rangle$

$\langle 3, \_, \text{kapp } ((\text{clo}(C, \emptyset[x \mapsto 1])), \_, (), \text{kret} \rangle$

$\langle D, \emptyset[x \mapsto 1][y \mapsto 3], \text{kret} \rangle$

$\langle 3, \_, \text{kapp } ((+, 1)\ \_\ ()\ \text{kret} \rangle$

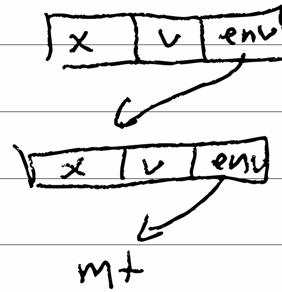
$\langle 4, \_, \text{kret} \rangle \longrightarrow 4$

6-13

naive , flat , nested

naive

closure	lambda	env



let y = 3 in
let z = 4 in
λ (x) (+ x y)

$\Downarrow$

flat

λ, (+ $\hat{0}$ $\hat{1}$)

ω	3

closure	lambda	vetor (values)

state = (nat, nat)

which

how many envs to go back

env = $\downarrow$ env , vector v