15-1/ $C_2 \to C_3$

exp := ... | (fun-ref label)
| (call arg arg ...)
tail := ... | (tailcall arg arg ...)
def := (define (label [var : ty] ...) : ty
info [label → tail] ...)
prog := (program info [defs ...] label)

---

econ : $R_4 \to C_3$

econ (rp ; ds (app (fun-ref main))) :=
(cp ; (map econd ds) main)

econd (define f ; rty [arg-name : ty] .. e) :=
(define (f [arg : ty] ..) rty (l → t'))
where l → t, t := econe e
l → t' := l → t [body → t]

econe (let [x := (app rator rands)] in body)
(set! x := (call rator rands) ; econe body)
econe (app rator rands) :=
(tailcall rator rands)

---

old : uncover-locals operated on progs
now: funs

15-2) $X_2 \rightarrow X_3$

arg := .... | (fun-ref label)

instr := .... | (indirect-callq arg) | (tailjmp label)

↓ callq *arg

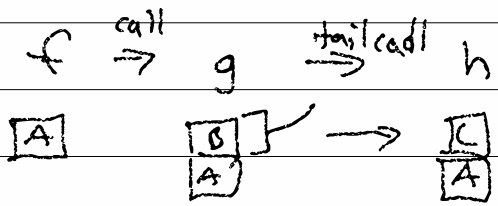
↓ postlude of fn
jmp label

START: alloc stack space (subq %rsp, $8)
save callee regs
jmp BODY

BODY: jmp END

the postlude

END: dealloc stack (addq %rsp, $8)
restor callee regs
retq



def := (define (label) info [label -> block] ...)

Prog := (xprogram i (def ...) label)

_main: init-gc
callq RMAIN
retq

RMAIN: START: ....
BODY: work
END: retq

15-3 select : $C_3 \Rightarrow X_3$

select$_p$ (program i ds) = (xprogram i (map select$_d$ ds))

select$_d$ (define i (f [a : ty]...) :ty [label ⇒ tail]... main)
:= (define i (f) label ⇒ block' start-lab)

where (label ⇒ block) := map select$_t$ label ⇒ tail

label ⇒ block' := label ⇒ block [START ↦
(block ∅ (movq r0 a0)
...
(movq r5 a5)

r = < rdi rsi rdx
rcx r8 r9 >

C) (define (f x y) ...) ⇒ (define (f)
X) START: [ ]
movq %rdi, !x
movq %rsi, !y
jmp BODY

(set! x
(f 1 2)) ⇒
movq $1, %rdi
movq !z, %rsi
callq f
movq %rax, !x

select$_e^{dst}$ (call rator rands) := ① ; callq (select$_a$ rator);
① = movq rands$_0$ r0
...
rands$_5$ r5
movq %rax, dst
tail: ① ; tail jmp (select$_a$ rator)

15-4 / register-alloc

conflict tail-jmp ⇒ don't read or write anything

callq ⇒ read all of regs (caller-saved)
and (callee if necessary)

assign / while replacing %x with %ordi
also replace (tail jmp label)
with ...
cleanup-code ; (jmp label)
↙

addq vector-var-count, ROOT-STACK-REG
addq local-var-cnt, RSP
popq all callee saves
popq %bp