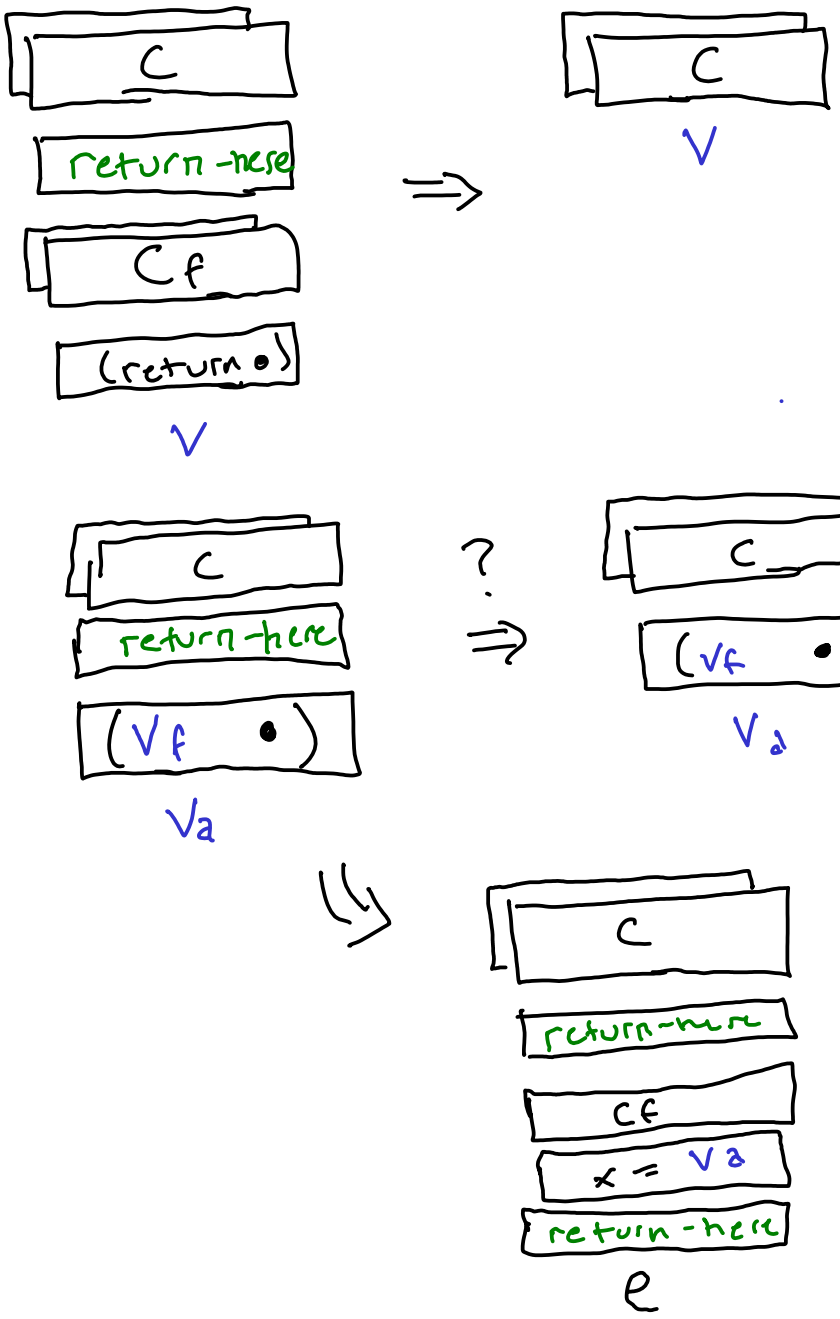


Example of non-tail call, needs to remember (if • ...) context.

```
(if (f x)
    (return 0)
    (return 1))
```

This is OK! (tail calls)

(if • (return 0) (return 1))

(f x)

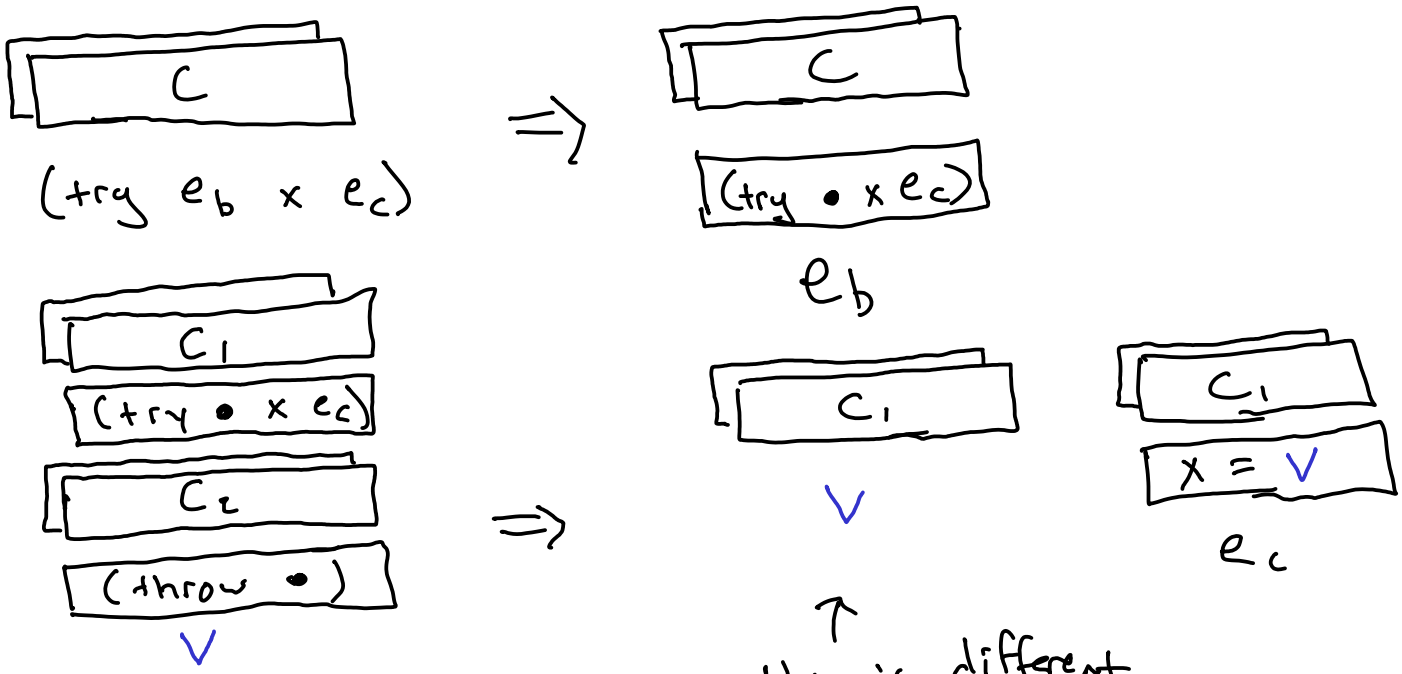
(try e_b x e_c)

(throw e)

(try (+ 1 (throw "err")) x (++ "msg was: " x))

(try (+ 1 2) x x) → 3

→ "msg was: err"



(gen e)

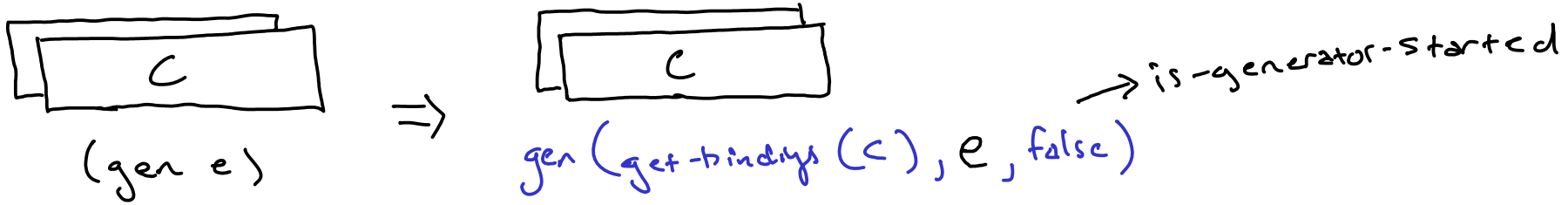
(next c)

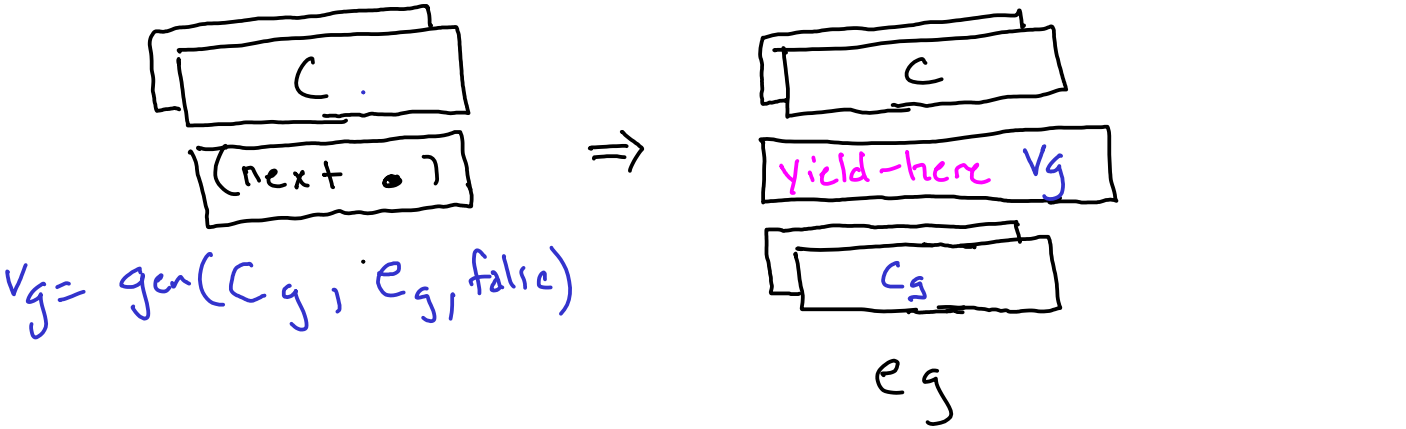
(yield e)

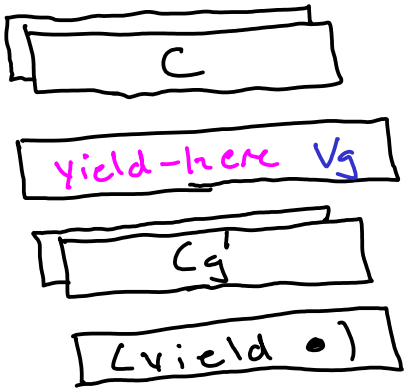
```
def nats(n)
    (gen
        i=0
        while True
            (yield i)
            i = i+1 )

g = nats()
(next g) ...
```

```
def nats():
  i = 0
  while True:
    yield i
    i = i + 1
```







V

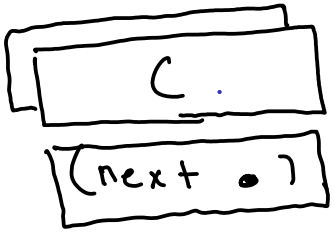
$\Rightarrow$

V

$v_g := gen(C_g', e, true)$

state update on v_g

(v_g before update is $gen(C_g, e, \{true\ or\ false\})$)



$v_g = gen(C_g, e_g, true)$

$\Rightarrow$

None

STACK COPYING ON YIELD