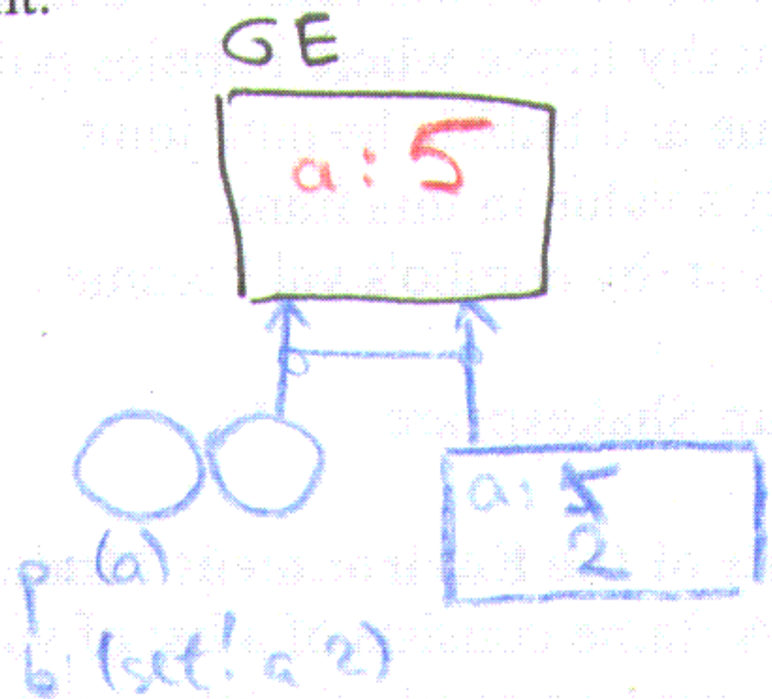
3. In A, bind the parameters of P to the argument values.
4. Evaluate the body of P with E as the current environment.



Examples:

```
(define a 5)
((lambda (a) (set! a 2)) a)
```

a ⇒ 5

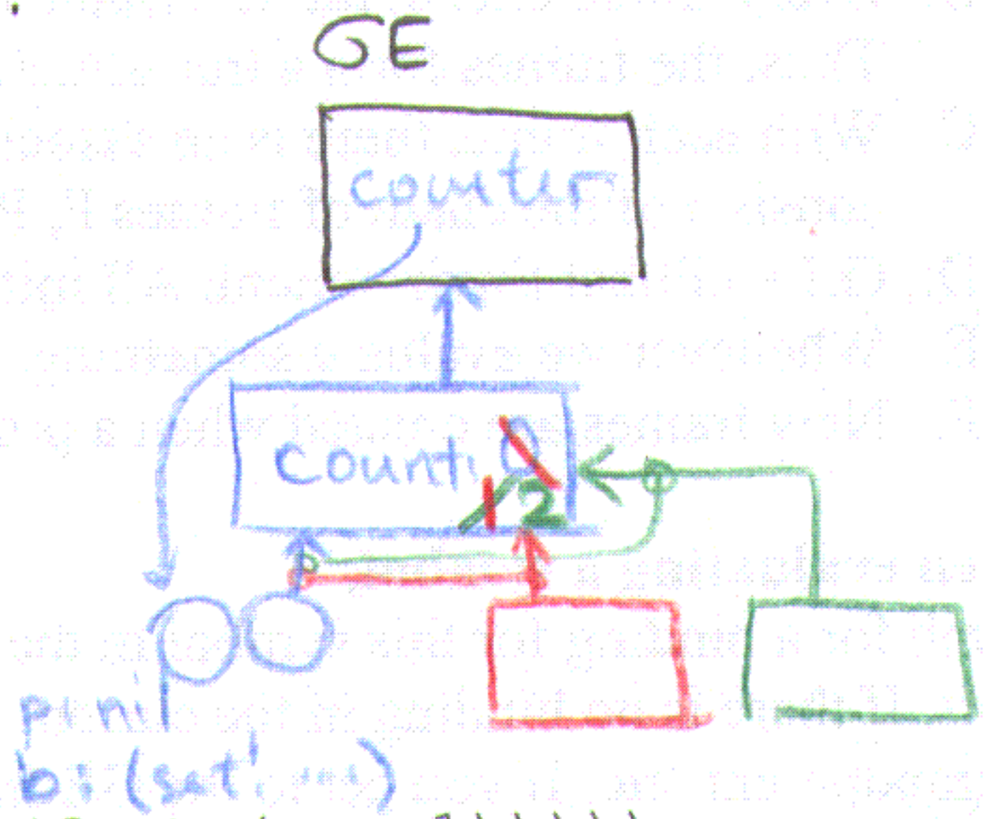
Note: same as a desugaring of

(let ((a a)) (set! a 2))

except with let, we don't draw the double bubble.

```
(define counter
   (let ((count 0))
      (lambda ()
         (set! count (+ 1 count))
         count)))
```

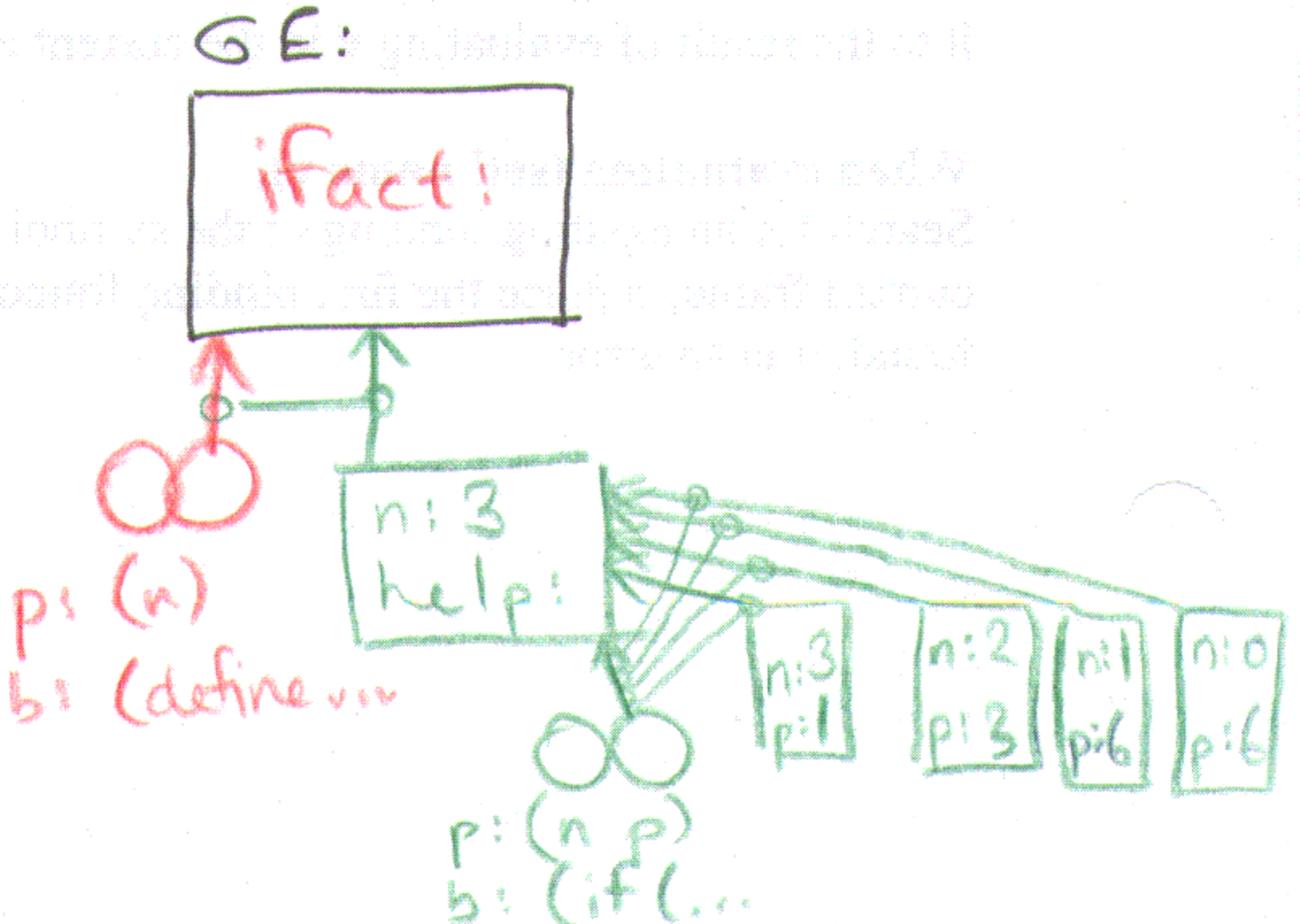
(counter) ⇒ 1
(counter) ⇒ 2




```
(define (fact n) (if (= n 0) 1 (* n (fact (- n 1)))))
(fact 3)
```

<omitted>

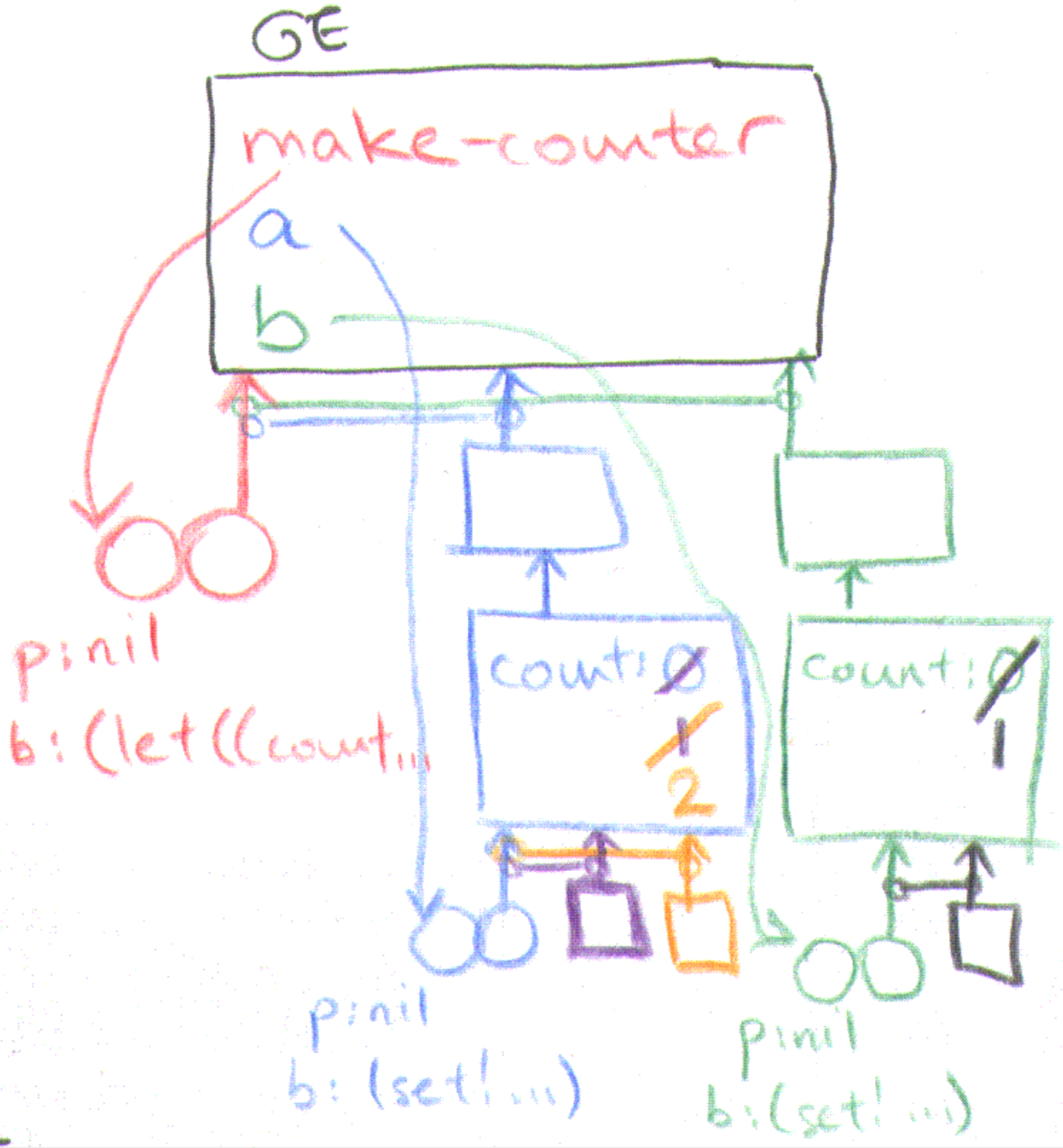
```
(define ifact
  (lambda (n)
    (define (help n p)
      (if (= n 0) p
       (help (- n 1) (* p n))))
    (help n 1)))
(ifact 3) ⇒ 6
```



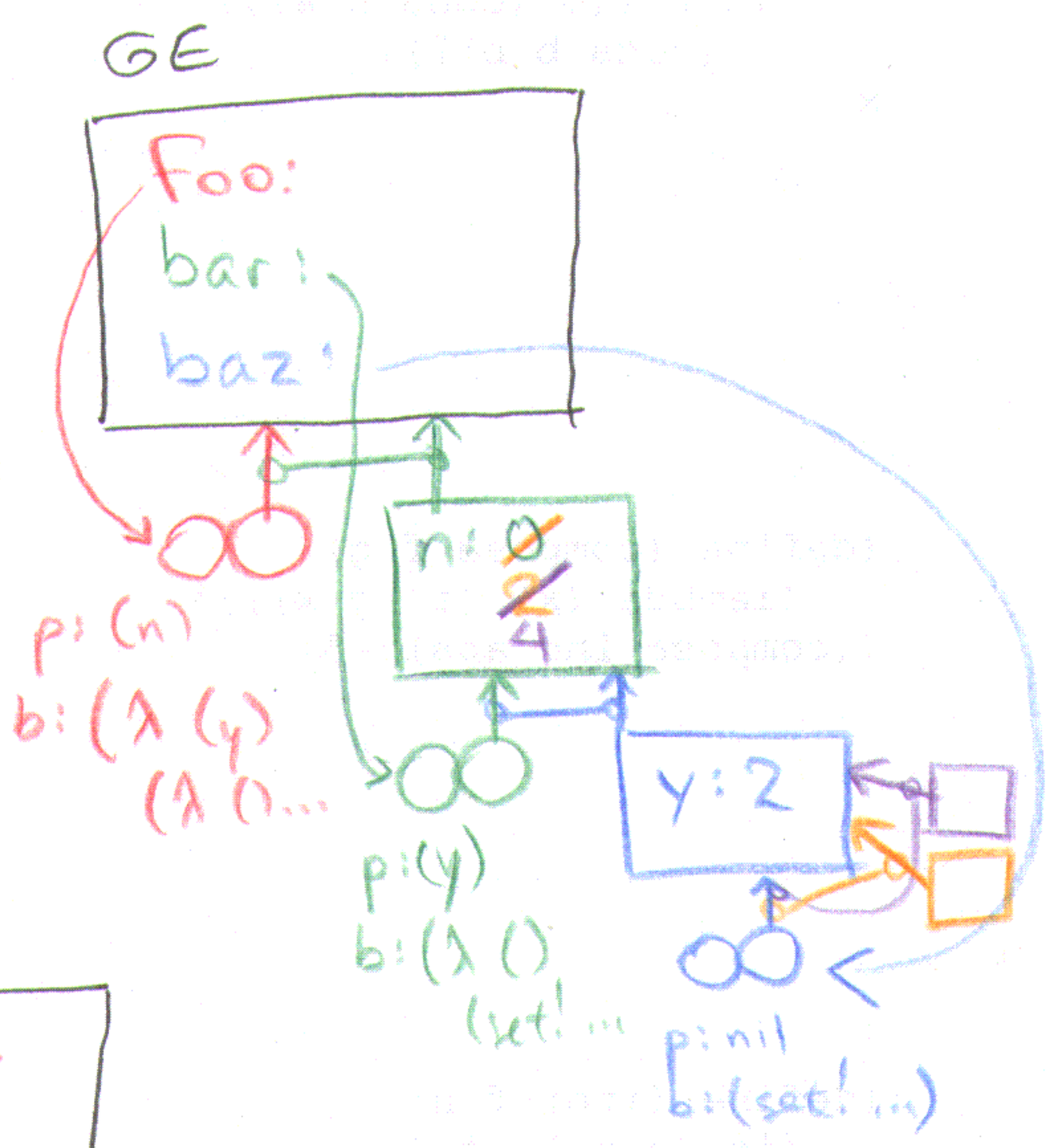
```
(define (make-counter)
   (let ((count 0))
      (lambda ()
         (set! count (+ 1 count))
         count)))
(define a (make-counter))
(define b (make-counter))
```

(a) ⇒ 1
(a) ⇒ 2
(b) ⇒ 1

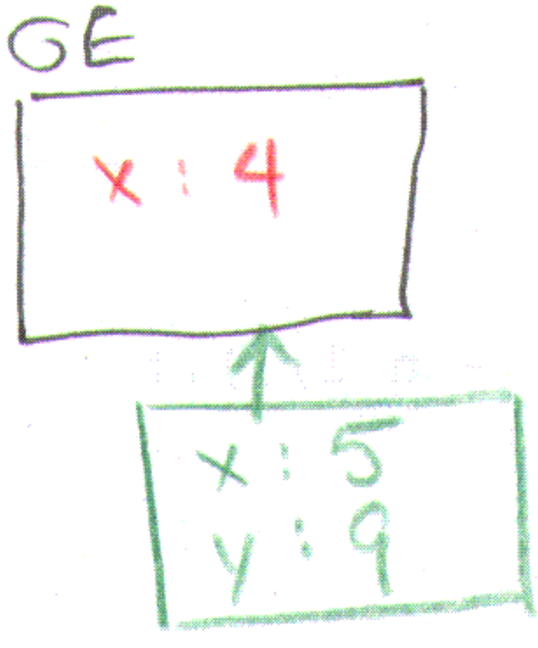
note that calls to make-counter, a, and b, drop empty frames since there are no parameters



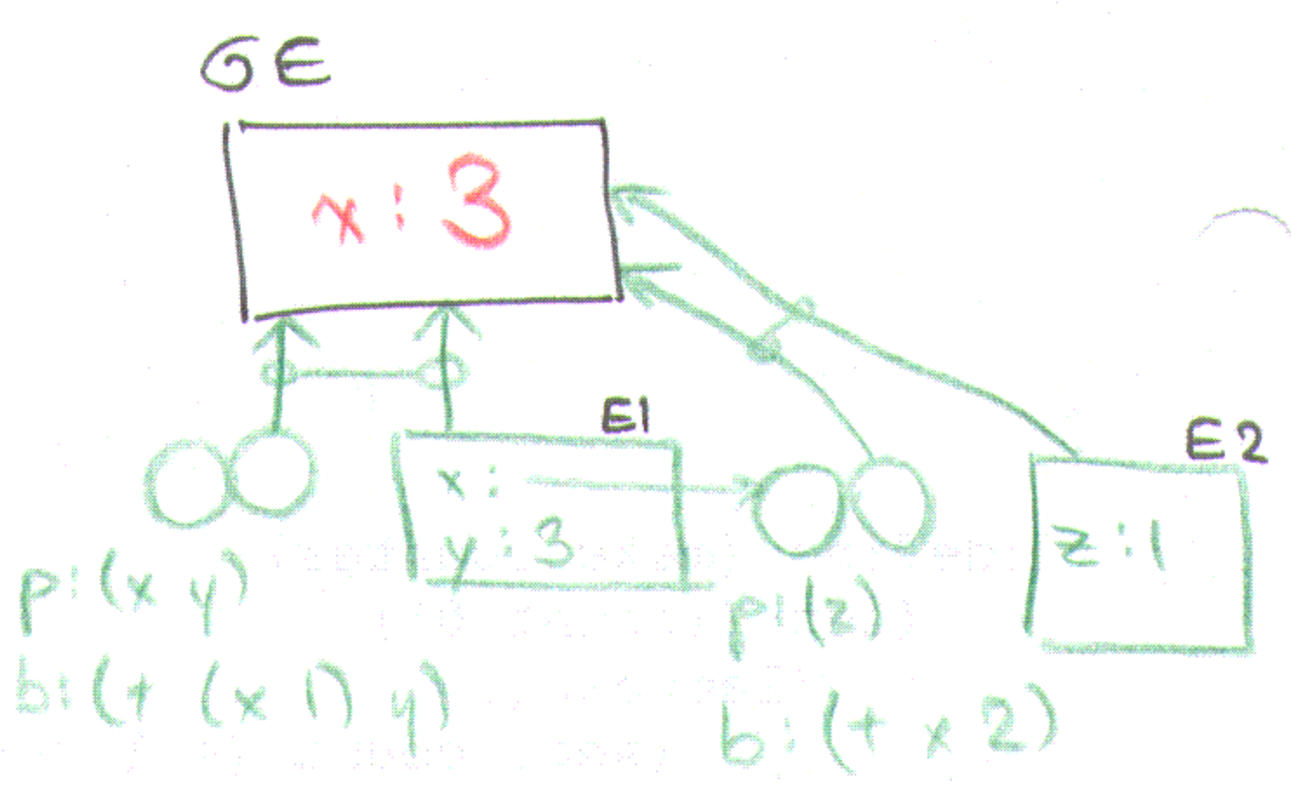
```
(define (foo n)
  (lambda (y)
    (lambda ()
      (set! n (+ n y))
      n)))
(define bar (foo 0))
(define baz (bar 2))
(baz) ⇒ 2
(baz) ⇒ 4
```




```
(define x 4)
(let ((x 5)
      (y (+ x 5)))
  (+ x y)) ⇒ 14
```



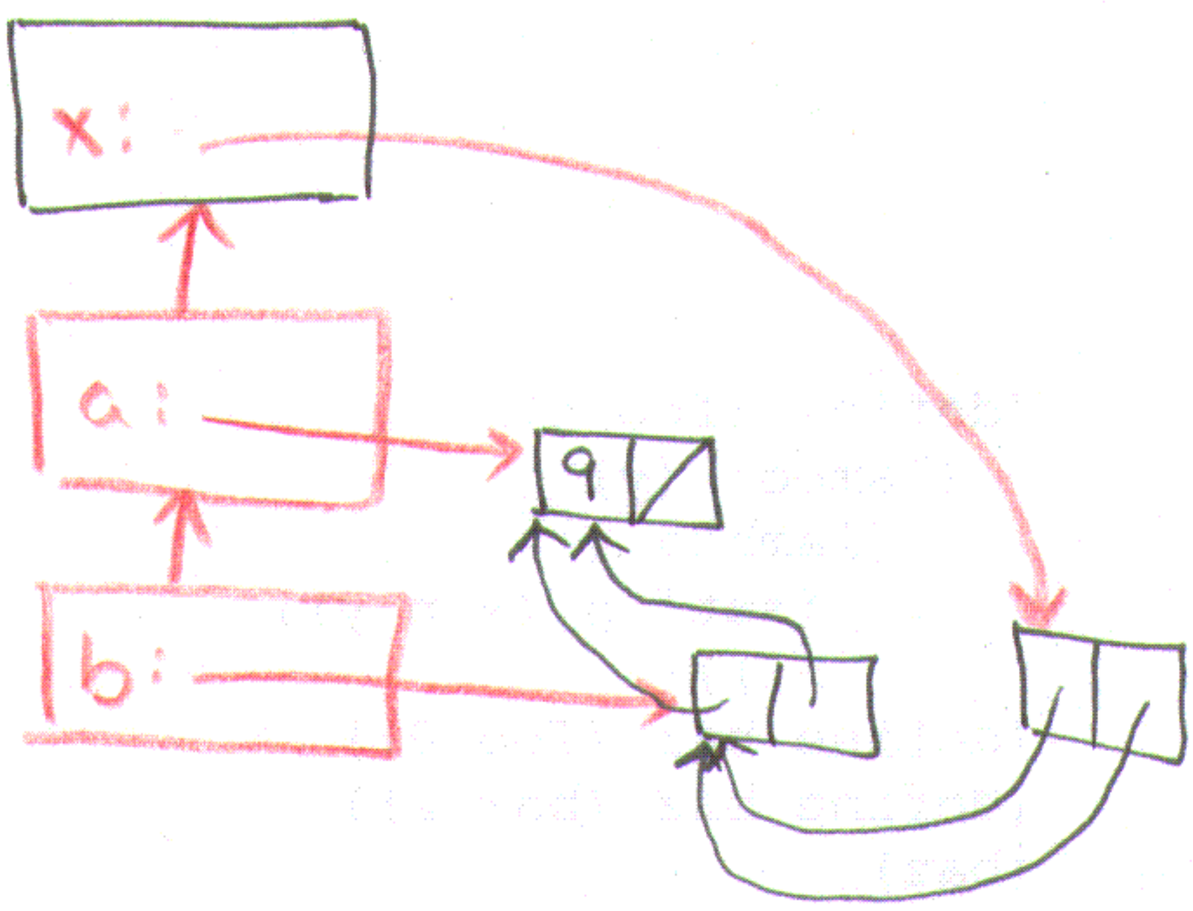
```
(define x 3)
((lambda (x y) (+ (x 1) y))
 (lambda (z) (+ x 2))
 3) ⇒ 8
```



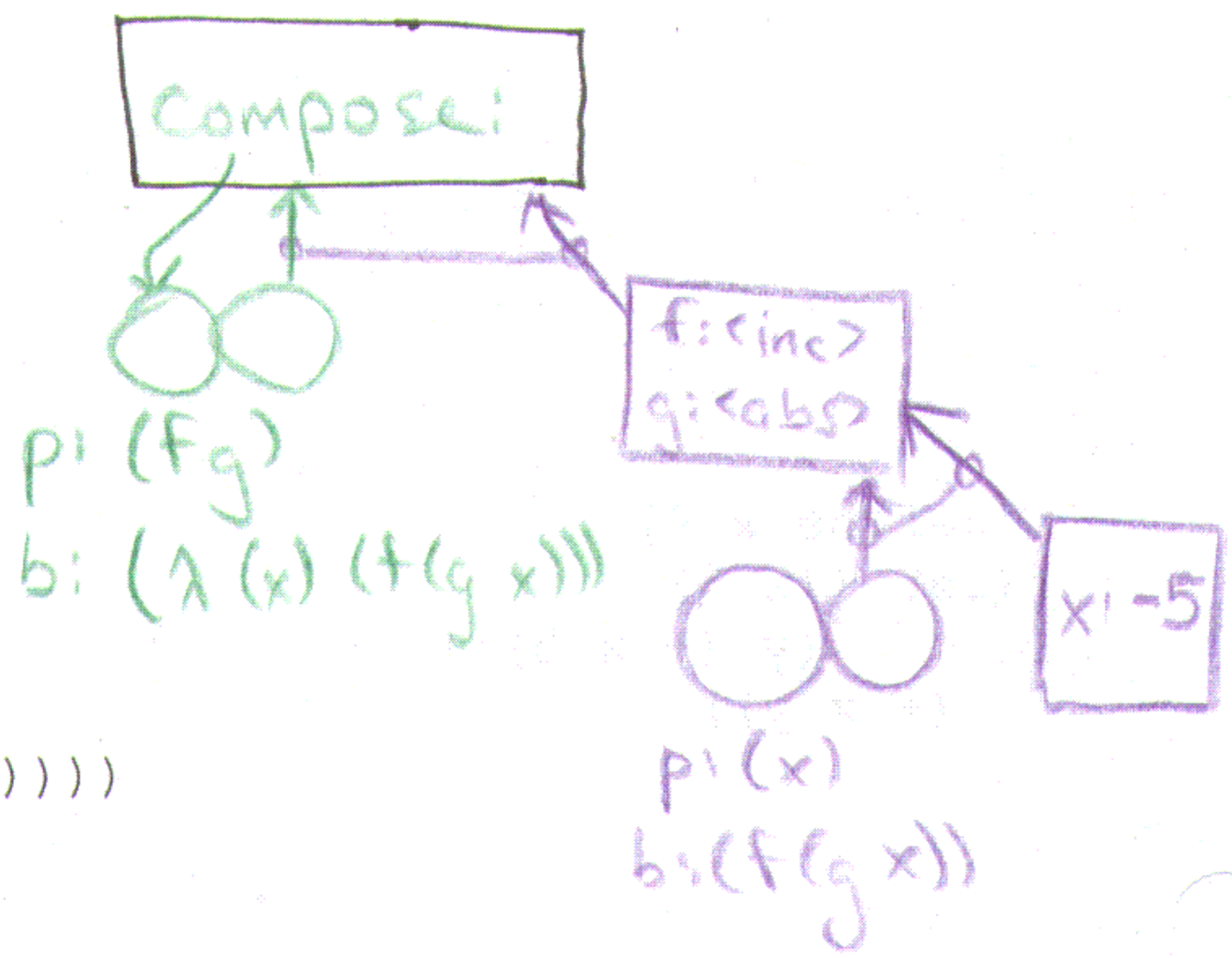
$$(x\ 1)\big|_{E1} \Rightarrow (+\ x\ 2)\big|_{E2} \Rightarrow 5$$

```
(define x
  (let ((a (list 9)))
    (let ((b (cons a a)))
      (cons b b))))
```

x ⇒ (((9) 9) (9) 9)




```
(define (compose f g)
   (lambda (x) (f (g x))))
((compose inc abs) -5) ⇒ 6
```




```
(define (iter f n)
   (if (= n 1) f
       (compose f (iter f (- n 1)))))
(define plus3 (iter inc 3))
(plus3 4)
```

<omitted>