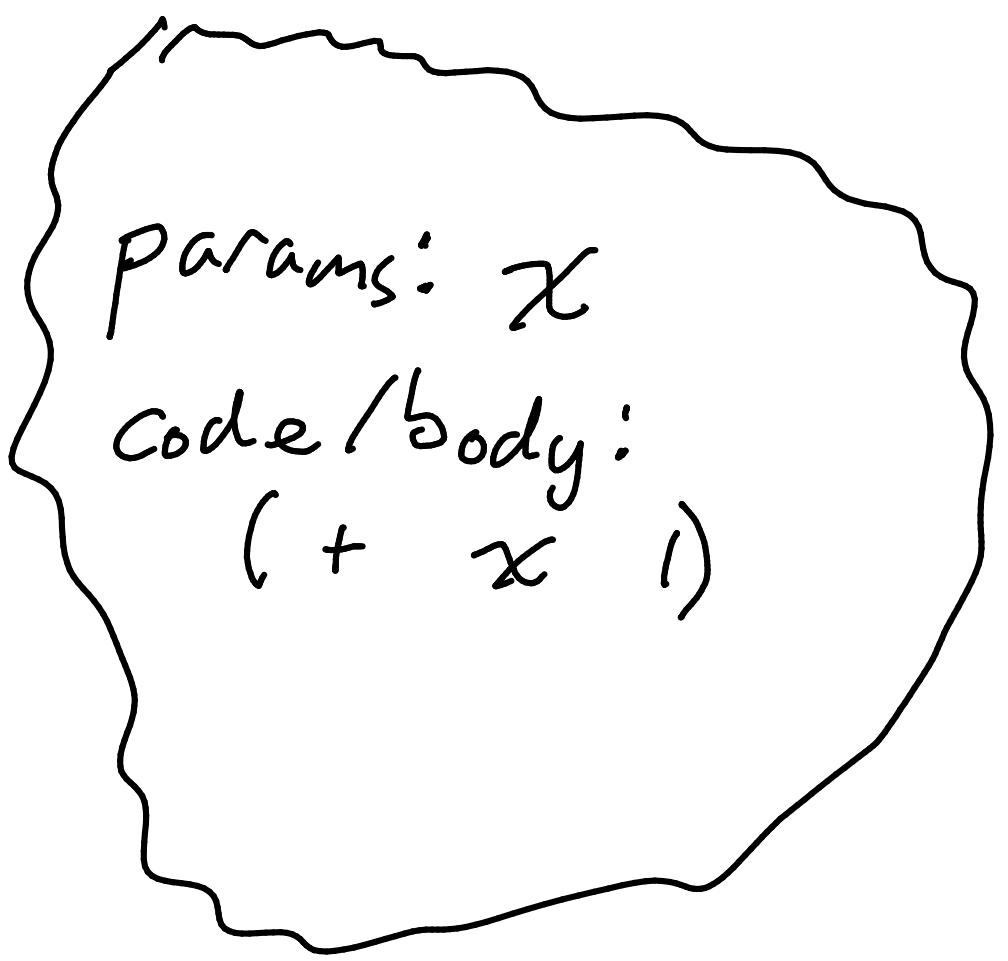
# Lambda and closures

What does lambda actually do?

```
(lambda (x)
  (+ x 1))
```

→ function

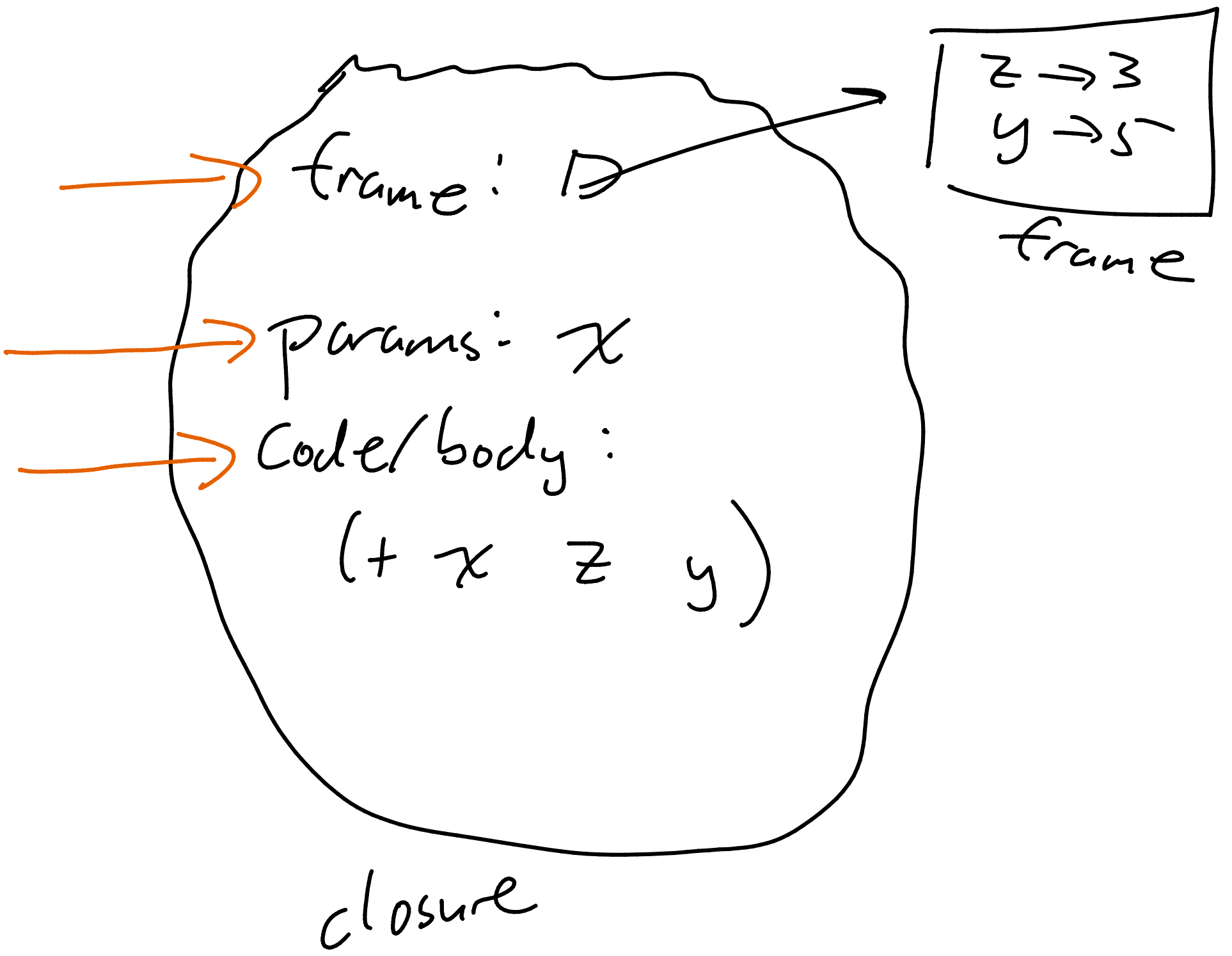
"closure"




closure struct

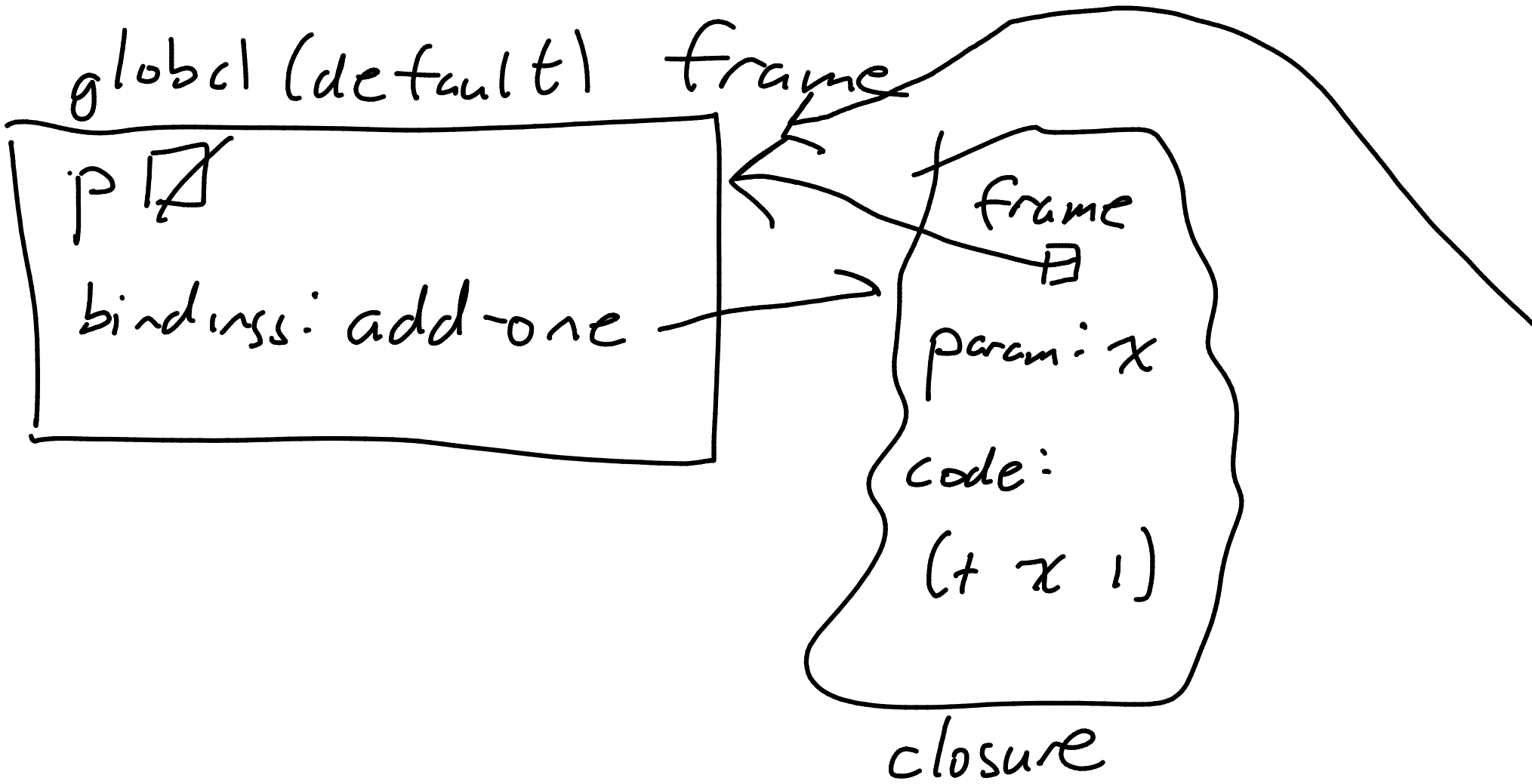
Another function

```
(let ((z 3) (y 5))
  (lambda (x)
    (+ x z y)))
```



define - adds a variable and its value to whichever frame is active (no new frame)

```
(define add-one
  (lambda (x)
    (+ x 1)))
```



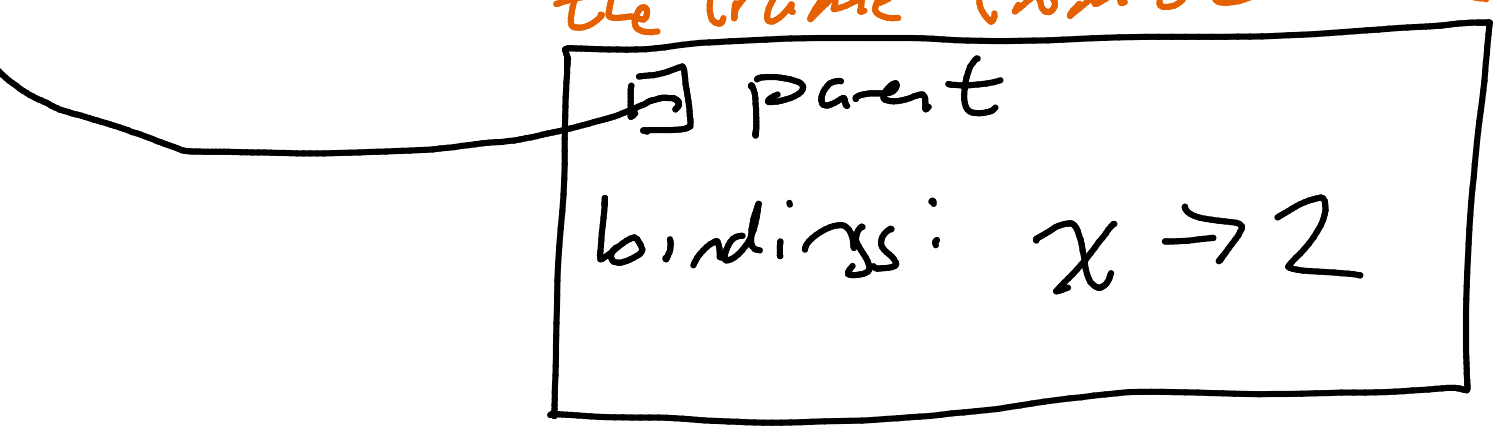
```
(define add-one
  (lambda (x)
    (+ x 1)))

(add-one 2)
```

Execute the function that <u>add-one</u> refers to. "Apply"

① Creates a new frame to hold parameters

② Connect parent ptr to the frame the closure pointed to

the frame from the closure

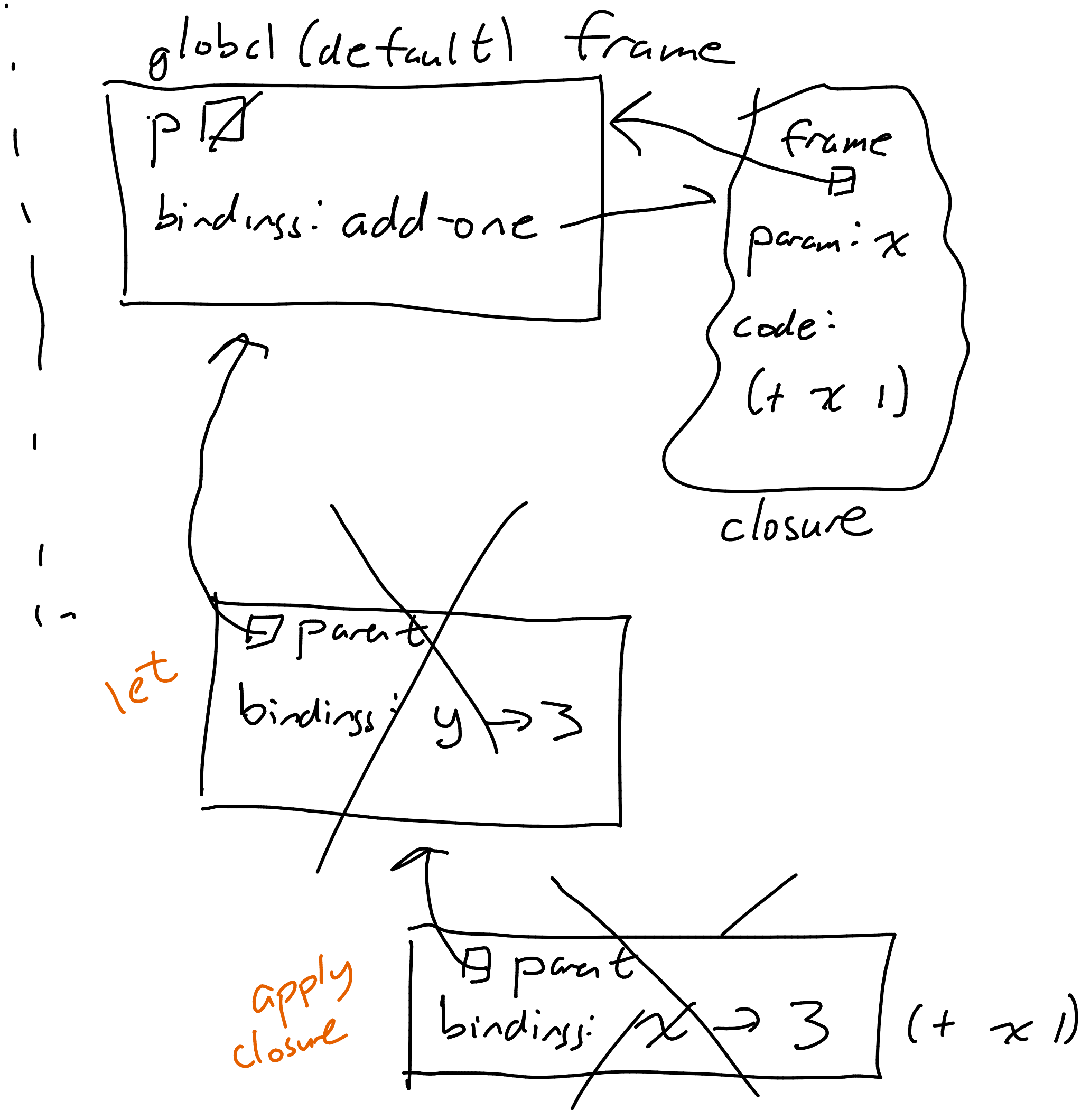



frame

③ Then run the code w/ the new frame as the active frame

```
(let ((y 3))
  (add-one y))
```

→ 4



(set! x 3)

"bang"

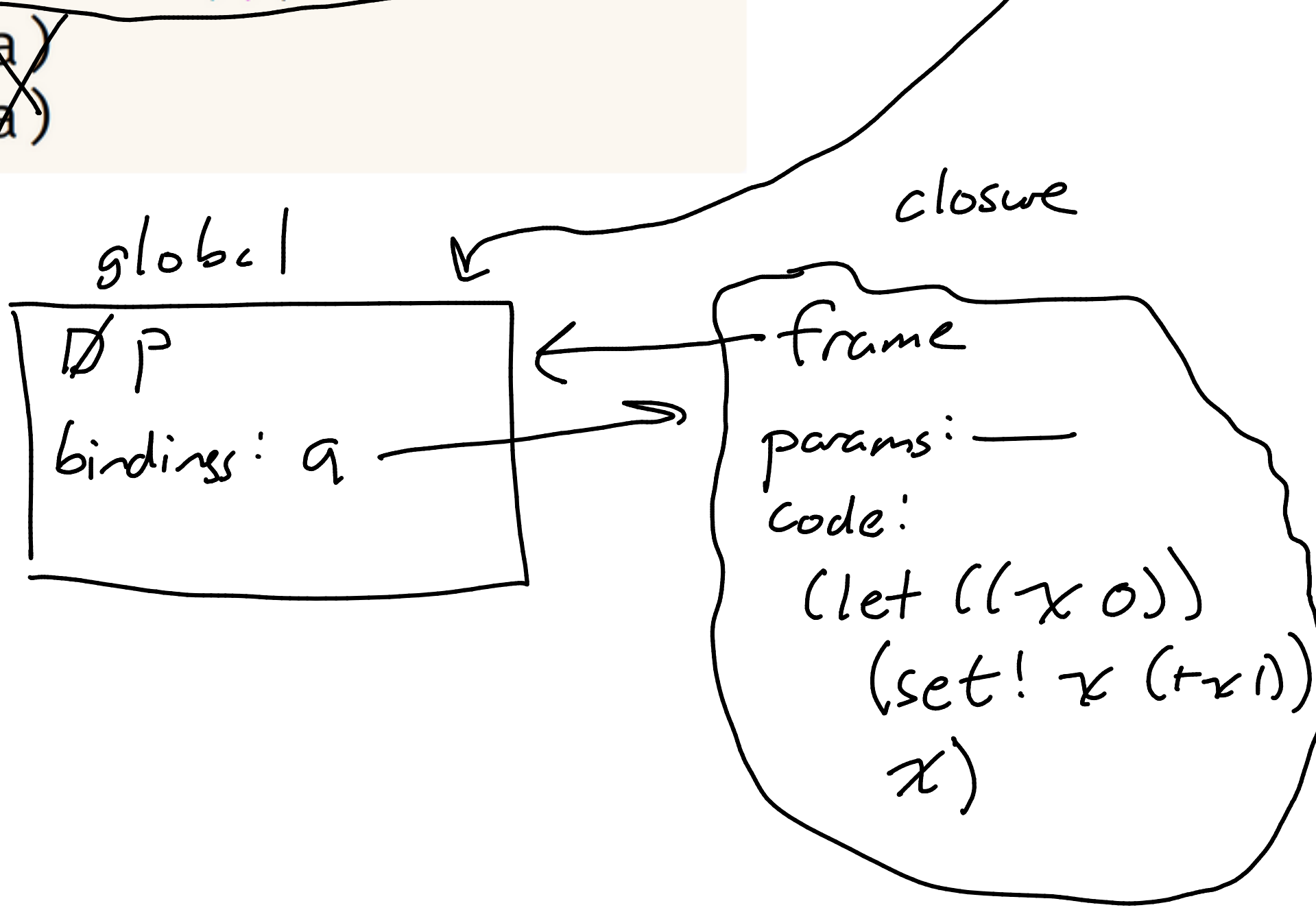
"shriek"

rebinds existing variable x in current frame to 3

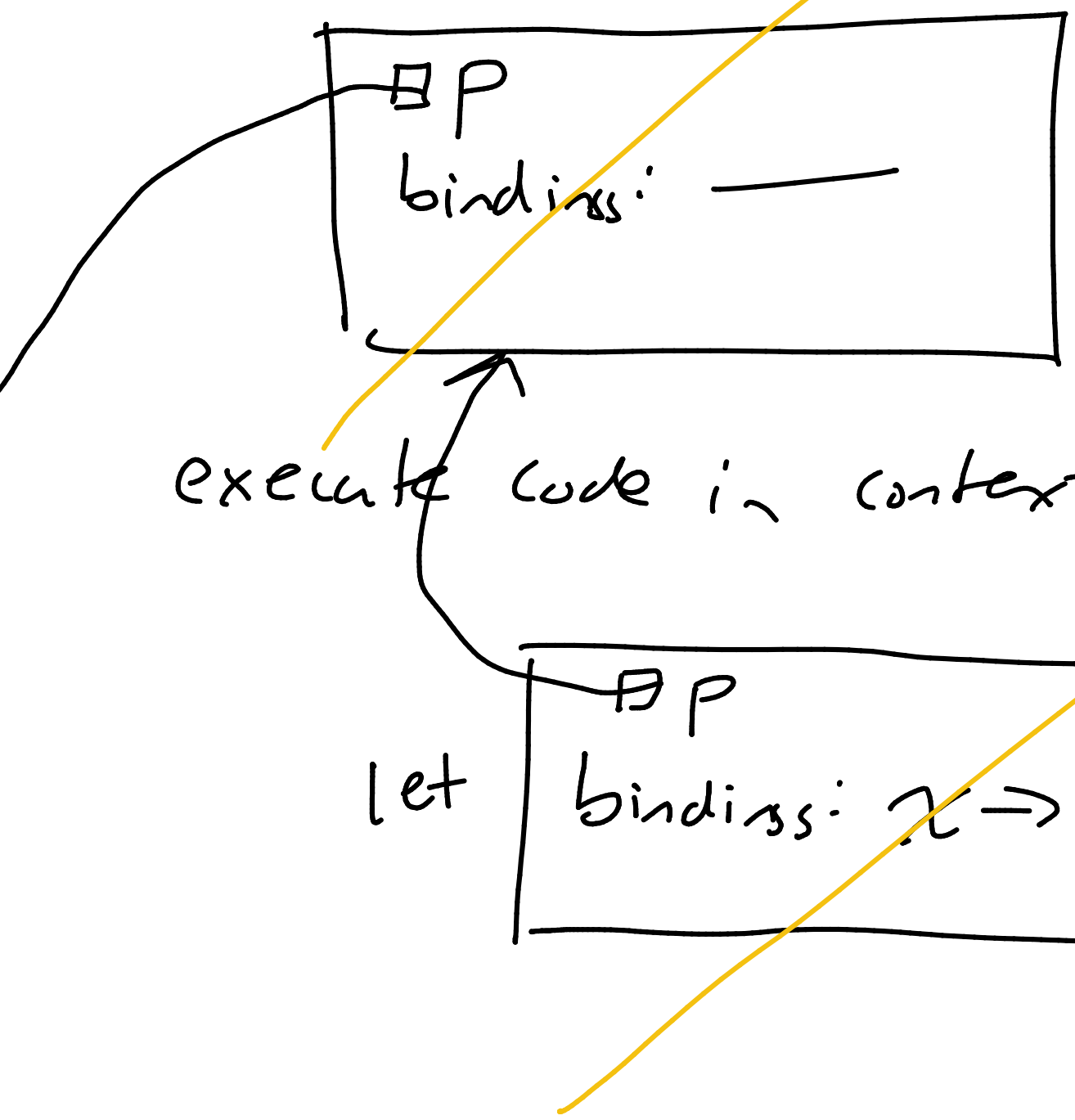
```
(define a
  (lambda ()
    (let ((x 0))
      (set! x (+ x 1))
      x)))
```

~~(a)~~

~~(a)~~



(a) "apply the function"




return

→ 1

```
(define a
  (let ((x 0))
    (lambda ()
      (set! x (+ x 1))
      x)))
(a)
(a)
```