Scope and Lexical Address

```
> (define x                    ; call this x1
    (lambda (x)                ; call this x2
      (map
         (lambda (x)           ; call this x3
           (+ x 1))            ; refers to x3
         x)))                  ; refers to x2

> (x '(1 2 3))                 ; refers to x1
(2 3 4)
```

Scope and Lexical Address

- Region inside a pair of parens is called a *block*.

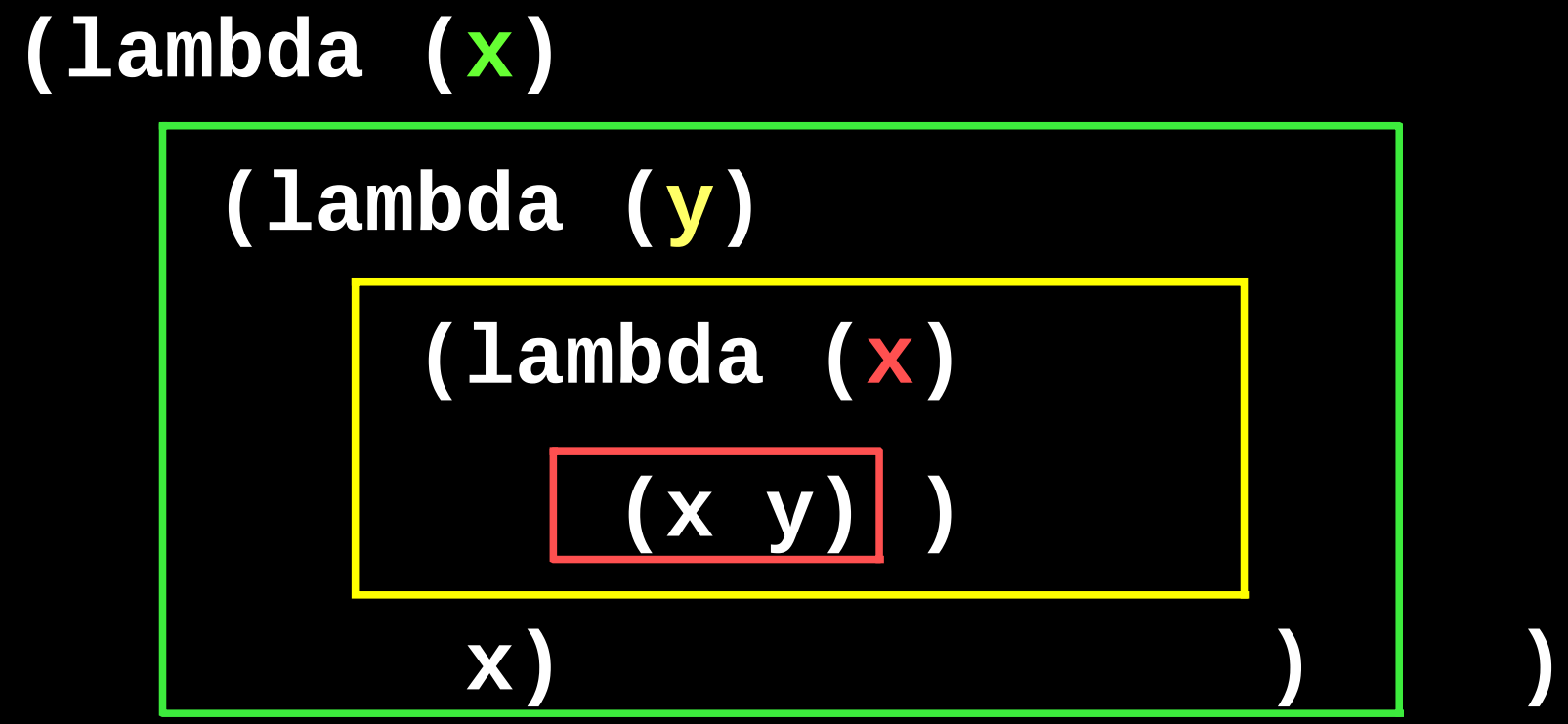
- Languages with (nested) blocks are called

block-structured

- *Lexical binding*: binding comes from most recent declaration

Scope and Lexical Address

Contour diagrams help (?) visualize block structure:



Scope and Lexical Address

Lexical address represents vars as (<depth> <position>):

```
(lambda (x y)
  (lambda (a)
     (x (a y)))
   x)
 (lambda (x y)
   (lambda (a)
      (x:1 0) ((a:0 0) (y:1 1))))
    (x:0 0))
```

Scope and Lexical Address

- So we can even get rid of names entirely!
- Similar to what a compiler does

```
(lambda 2    ; two args
  ((lambda 1 ; one arg
    ((:1 0) (:0 0) (:1 1))))
     :0 0)))
```