Clicker: If using pass by value, what’s the value of a at the end of the program?

```
void sub(int x) {
    x = x + 5;
}
a = 10;
sub(a);
```

A. 15

[✓ B. 10]

C. 20

Clicker: If using pass by value-result, what’s the value of a at the end of the program?

```
void sub(int x) {
    x = x + 5;
}
a = 10;
sub(a);
```

[✓ A. 15]

B. 10

C. 20

Clicker: how should “3+4” be written in Scheme?

[✓ A. (+ 3 4)]

B. (+ (3) 4)

C. (+ 3 (4))

D. (+ (3) (4))

E. All of the above are okay

Clicker: in Scheme, “$f(x) = x$” as an anonymous fun is written as

A. (lambda (x) (x))

[✓ B. (lambda (x) x)]

Clicker: in Scheme, “$f(x) = x*x$” as an anonymous fun is written as

[✓ A. (lambda (x) (* x x))]

B. (lambda (x) (* (x) x))

Clicker: In Scheme, “(3+8)+2” is written as

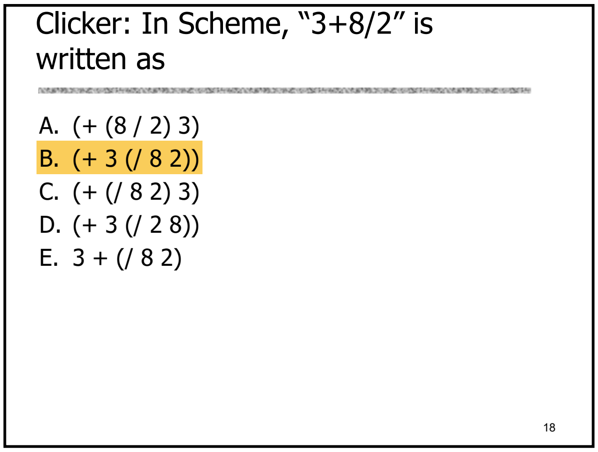
A. (+ (3 + 8) 2)

B. (+ 2 (+ 3 8))

[✓ C. (+ (+ 3 8) 2)]

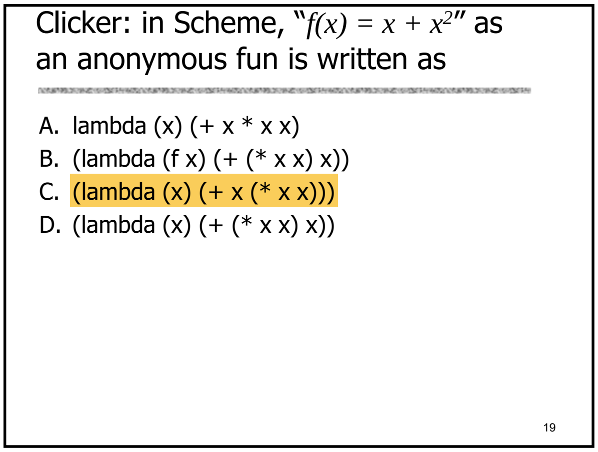
D. + (+ 3 8) 2

E. (+ + 3 8 2)



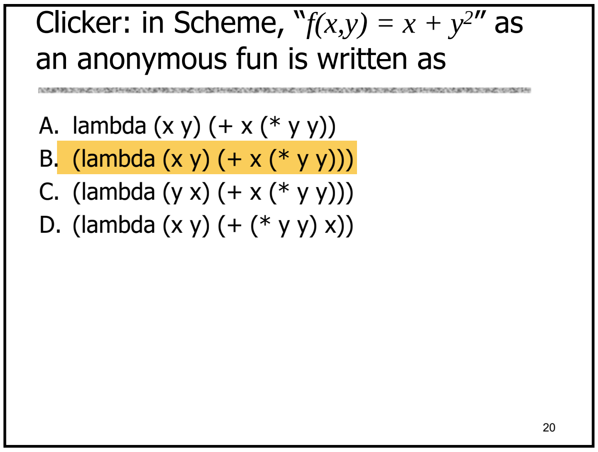
Clicker: In Scheme, “3+8/2” is written as

A. (+ (8 / 2) 3)
B. (+ 3 (/ 8 2))
C. (+ (/ 8 2) 3)
D. (+ 3 (/ 2 8))
E. 3 + (/ 8 2)



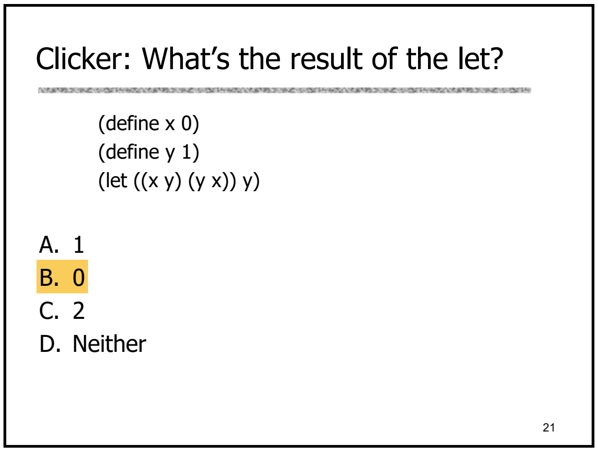
Clicker: in Scheme, “$f(x) = x + x^2$” as an anonymous fun is written as

A. lambda (x) (+ x * x x)
B. (lambda (f x) (+ (* x x) x))
C. (lambda (x) (+ x (* x x)))
D. (lambda (x) (+ (* x x) x))



Clicker: in Scheme, “$f(x,y) = x + y^2$” as an anonymous fun is written as

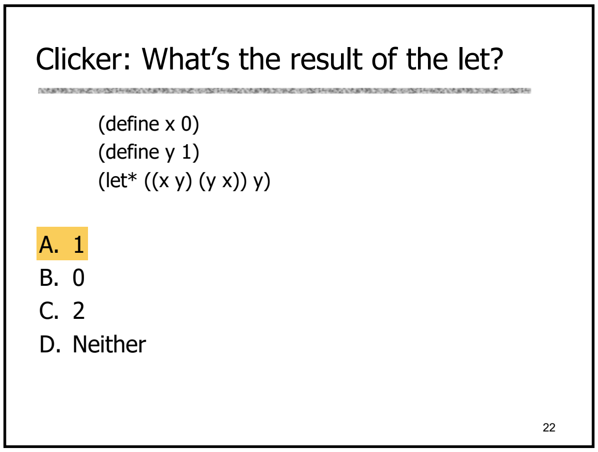
A. lambda (x y) (+ x (* y y))
B. (lambda (x y) (+ x (* y y)))
C. (lambda (y x) (+ x (* y y)))
D. (lambda (x y) (+ (* y y) x))



Clicker: What’s the result of the let?

```
(define x 0)
(define y 1)
(let ((x y) (y x)) y)
```

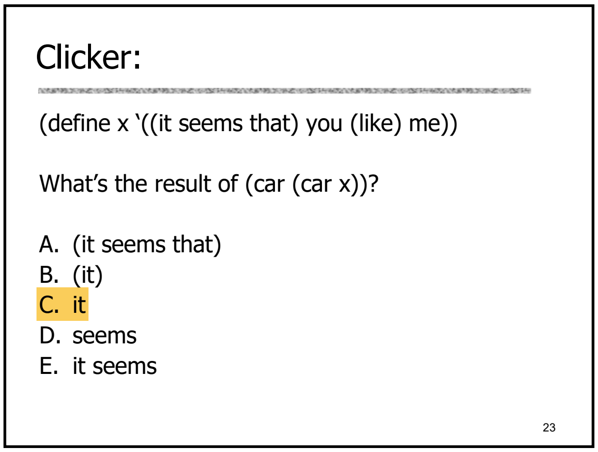
A. 1
B. 0
C. 2
D. Neither



Clicker: What’s the result of the let?

```
(define x 0)
(define y 1)
(let* ((x y) (y x)) y)
```

A. 1
B. 0
C. 2
D. Neither



Clicker:

(define x ‘((it seems that) you (like) me))

What’s the result of (car (car x))?

A. (it seems that)
B. (it)
C. it
D. seems
E. it seems

Clicker:

(define x ‘((it seems that) you (like) me))

What’s the result of (cdr (cdr x))?

A. you
B. (like me)
C. ((like) me)
D. (like) me
E. (like)

Clicker:

What’s the result of (cons ‘a ‘())?

A. a
B. (a)
C. ((a))
D. ()
E. None of the above

Clicker:

What’s the result of (cons ‘a (cdr ‘((b) c d)))?

A. (a (b) c d)
B. (a (c d))
C. (a c d)
D. (a b c d)
E. None of the above

Clicker:

What’s the result of
(map (lambda (x) (list x (+ x 1))) ‘(3 7 12 9))?

A. (4 8 13 10)
B. (3 7 12 9)
C. (3 4 7 8 12 13 9 10)
D. ((3 4) (7 8) (12 13) (9 10))
E. None of the above

Clicker:

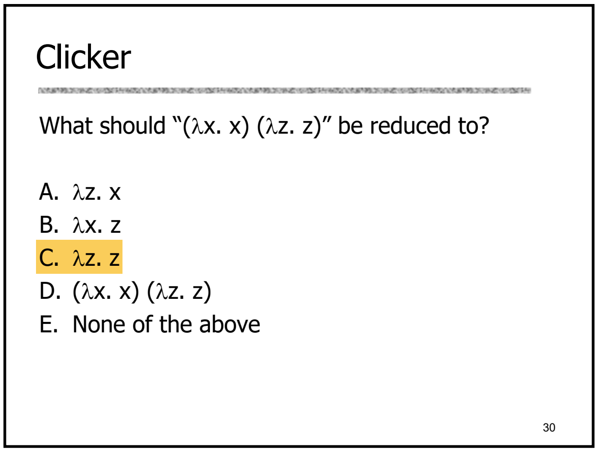
What’s the result of
(map length ‘((a) (a b) (a b c) ()))?

A. (1 2 3 0)
B. ((1) (2) (3) (0))
C. 4
D. (1 1 1 1)
E. None of the above

Clicker:

What’s the result of
(reduce (lambda (x y) (and x y)) ‘(#t #f #t) #t)?

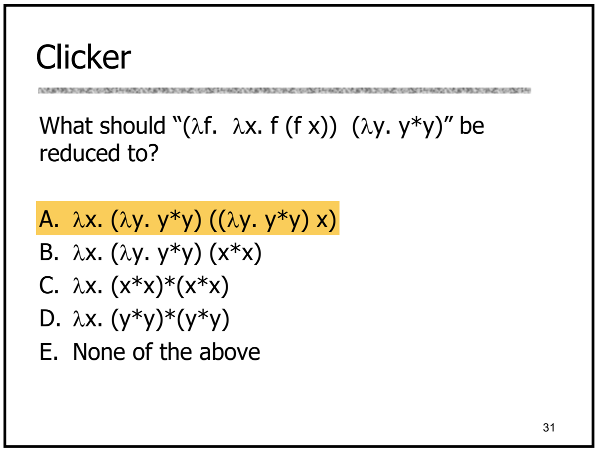
A. #t
B. #f
C. (#t #f #t #t)
D. Runtime error
E. None of the above



Clicker

What should “(λx. x) (λz. z)” be reduced to?

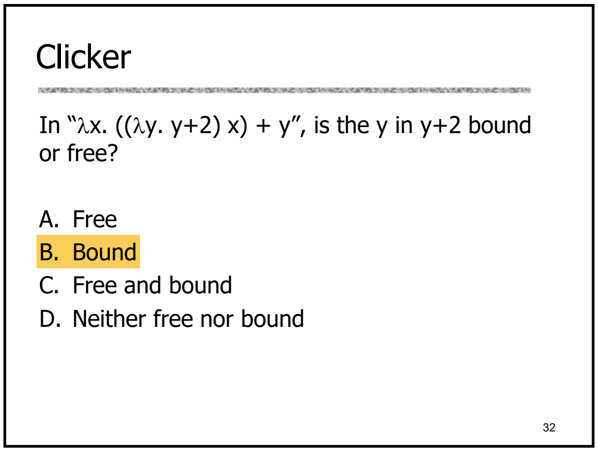
A. λz. x
B. λx. z
C. λz. z
D. (λx. x) (λz. z)
E. None of the above



Clicker

What should “(λf. λx. f (f x)) (λy. y*y)” be reduced to?

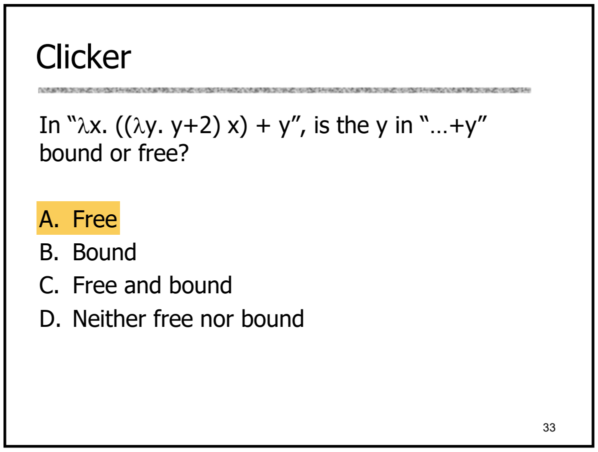
A. λx. (λy. y*y) ((λy. y*y) x)
B. λx. (λy. y*y) (x*x)
C. λx. (x*x)*(x*x)
D. λx. (y*y)*(y*y)
E. None of the above



Clicker

In “λx. ((λy. y+2) x) + y”, is the y in y+2 bound or free?

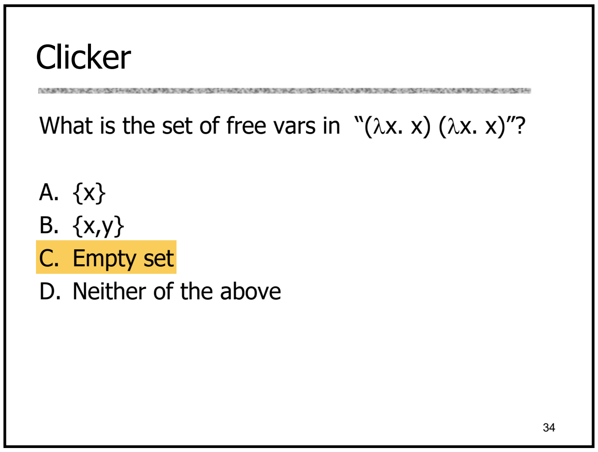
A. Free
B. Bound
C. Free and bound
D. Neither free nor bound



Clicker

In “λx. ((λy. y+2) x) + y”, is the y in “…+y” bound or free?

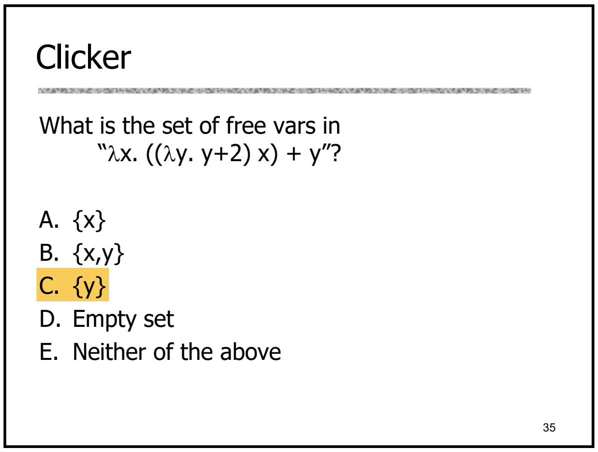
A. Free
B. Bound
C. Free and bound
D. Neither free nor bound



Clicker

What is the set of free vars in “(λx. x) (λx. x)”?

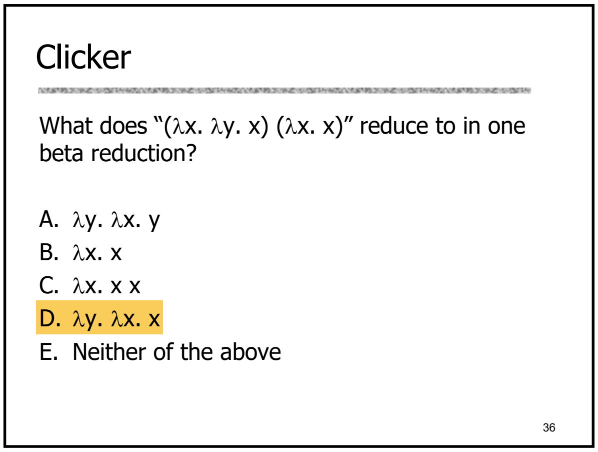
A. {x}
B. {x,y}
C. Empty set
D. Neither of the above



Clicker

What is the set of free vars in
“λx. ((λy. y+2) x) + y”?

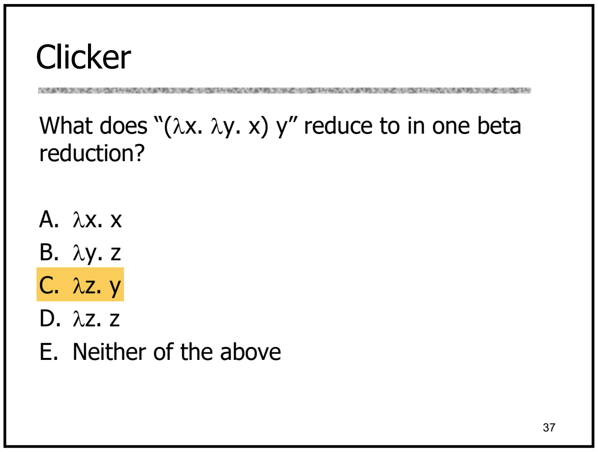
A. {x}
B. {x,y}
C. {y}
D. Empty set
E. Neither of the above



Clicker

What does "(λx. λy. x) (λx. x)" reduce to in one beta reduction?

A. λy. λx. y
B. λx. x
C. λx. x x
D. λy. λx. x
E. Neither of the above



Clicker

What does "(λx. λy. x) y" reduce to in one beta reduction?

A. λx. x
B. λy. z
C. λz. y
D. λz. z
E. Neither of the above