Clicker Questions
October 9
We are going to need a datatype for environments. Which Scheme expressions extend the environment?

A. let and calls
B. let and lambda
C. let and lambda and calls
D. None of the above are completely correct
Answer A: Let and calls
Consider the expression

(let ([x 3])
  (let ([x 4])
    (+ x 5)))

This is badly written, but is it illegal?

A. This is an invalid expression.
B. This is a valid expression that evaluates to 8.
C. This is a valid expression that evaluates to 9.
D. This is a valid expression that evaluates to 12.
Answer C: It evaluates to 9.
We want a data structure for extended environments. We might have expression \((\text{let } ([x \ 3] [y \ 5]) \ \text{body})\) or an expression like \(( (\text{lambda} (x \ y) \ \text{body}) \ 3 \ 5)\). Here are 4 possible structures:

A. \('(\text{extended-env} (((x \ 3) (y \ 5))) \ \text{old-environment})\)
   i.e., extend the environment with a list of (symbol value) pairs

B. \('(\text{extend-env} (((x \ 3) (y \ 5)))\)
   i.e., throw out the old environment and just use the new one

C. \('(\text{extended-env} (x \ y) '(3 5) \ \text{old-environment})\)
   i.e., extend the environment with a list of symbols and a list of values

D. \('(\text{extended-env} x \ 3 (\text{extended-env} y \ 5) \ \text{old-environment})\)
   i.e., extend the environment one symbol, value at a time
There is no one "right" answer here (though B is clearly a wrong answer). You could use A, C, or D; we are going to use (at least, I will recommend we use) C: a list of symbols, a list of values, and the old environment.
What does the following evaluate to?
(let ([a 3])
 (let ([f (lambda (x) (+ x a))])
   (let ([a 300])
     (f 5))))

A. This is an invalid expression
B. This is a reasonable expression that evaluates to 8.
C. This is a reasonable expression that evaluates to 305.
D. This is a valid but unreasonable expression and any self-respecting Scheme interpreter will refuse to evaluate it.
Answer B: This is a perfectly reasonable expression that evaluates to 8.