* Top of Form
* Evaluate (car (cdr (cdr '(salamander frog toad newt)))): **toad**
* Evaluate (cdr '(+ 10 15)): **(10 15)**
* Evaluate (cdr '((red) (blue green))): **((blue green))**
* Evaluate (\* 2 (\* 2 (\* 2 2))): **16**
* Evaluate (cdr '(yellow purple orange)): **(purple orange)**
* Evaluate (car (cdr '(one two three))): **two**
* Evaluate (+ 8 (- 3 2)): **9**
* Evaluate (cons (car '(pooh piglet)) (cdr '(eeyore tigger))): **(pooh tigger)**
* Evaluate (cdr '(oak maple)): **(maple)**
* Evaluate (cdr '(one two three)): **(two three)**
* Evaluate (cdr '(oak maple poplar elm)): **(maple poplar elm)**
* Top of Form
* Evaluate (car '(a (b) c)): **a**
* Evaluate (quote (+ 2 4)): **(+ 2 4)**
* Evaluate (quote (- 10 1)): **(- 10 1)**
* Evaluate (quote (car (quote (a b c)))): **(car (quote (a b c)))**
* Evaluate (\* (+ 3 10) (+ 1 1)): **26**
* Evaluate (cdr '(+ 10 15)): **(10 15)**
* Evaluate (sqrt 9): **3**
* Evaluate (- 2 5): **-3**
* Evaluate (car '(one two three)): **one**
* Evaluate (cons 'chair '(table sofa)): **(chair table sofa)**
* Evaluate (car (cdr (cdr '(one two three)))): **three**
* Evaluate (car (car '((gold) (red) (purple)))): **gold**
* Which of the expressions below evaluates to orange? **(cadar '((green orange) (blue red)))**
* Evaluate (and (> 5 3) (< 6 8)) **#T**
* (equal? 'glassboro '(glassboro)) **#F**
* Evaluate (car (cadr '((orwell 1984) (shakespeare hamlet) (homer iliad)))) **shakespeare**
* Evaluate (cons (cadr '(cup glass mug)) (cddr '(plate saucer))) **(glass)**
* Evaluate (cons '(crab) (cons '(oyster) '(lobster))) **((crab) (oyster) lobster)**
* (= '(salem gloucester camden) '(salem gloucester camden)) **This expression would produce an error message when evaluated**
* (eq? 'apple 'apple) **#T**
* Evaluate (cons (> 34 56) '(100)) **(#F 100)**
* (= 22 22) **#T**
* Evaluate (or (= 50 13) (> 16 88)) **#F**
* Evaluate (cons (car '(cup glass mug)) (cdr '(plate saucer))) **(cup saucer)**
* Evaluate (and (> 100 77) (not (= 50 13))) **#T**
* (eq? 22 22) **#T**
* Which of the expressions below evaluates to (b c)? **(cadr '((a d) (b c) (e f)))**
* (= 'glassboro 'glassboro) **This expression would produce an error message when evaluated**
* (equal? 'glassboro 'glassboro) **#T**
* Evaluate (cadr (cadr '((orwell 1984) (shakespeare hamlet) (homer iliad)))) **hamlet**
* Evaluate (cadr (car '((orwell 1984) (shakespeare hamlet) (homer iliad)))) **1984**
* Evaluate (cadr '(a b c d e)) **b**
* Scheme and Common Lisp are two dialects of Lisp that are widely used
  + Scheme is characterized by its small size, its exclusive use of static scoping, and its treatment of functions as first-class entities
  + Scheme functions can be assigned to variables, passed as parameters, and returned as the values of function applications; can also be elements of lists
  + Scheme is well suited to educational applications