

Solutions to Sample Quiz 1

Problem 1

11
error (arguments passed in wrong order)
36

Problem 2

Applicative order: Evaluate all subexpressions first, then apply the first to the rest.
(Scheme uses this.)

Normal order: No arguments are evaluated until they are needed. Fully expand,
then reduce.

In Scheme (applicative order), the following two items could be printed:

one two plus
two one plus

In normal-order Scheme, the following two items could be printed:

plus one two
plus two one

Problem 3

#f #f #t
#f #f #t
#f #f #t
#t #t #t

Problem 4

(caddr first-list)
(caadr second-list)

Problem 5

```
(define (merge list1 list2)
  (cond ((null? list1) list2)
        ((null? list2) list1)
        ((= (car list1) (car list2))
         (cons (car list1)
                (merge (cdr list1) (cdr list2))))
        (< (car list1) (car list2))
         (cons (car list1)
                (merge (cdr list1) list2)))
        (else (cons (car list2) (merge list1 (cdr list2))))))
```

Time: $\Theta(n)$

Space: $\Theta(n)$

n is dependent upon the size of the longer list

Recursive process

Problem 6

```
(define (apply-twice f)
  (lambda (x) (f (f x))))
```

Problem 7

```
(tree-manip test-tree
  0
  (lambda (x) x)
  car
  cdr
  +)

(tree-manip test-tree
  nil
  (lambda (x) (list x))
  car
  cdr
  append)
```

Problem 8

```
(define (item-name item)
  (caar item))

(define (item-value item)
  (cdar item))

(define (item-condition item)
  (cadr item))

(define first-item car)

(define rest-items cdr)

(define (total-value item-list)
  (if (null? item-list)
      0
      (+ (item-value (first-item item-list))
         (total-value (rest-items item-list)))))
```

Time: $\Theta(n)$

Space: $\Theta(n)$

n is dependent upon the length of the list of items

Recursive process

Problem 9

```
(define (car z)
  (z 'car))

(define (cdr z)
  (z 'cdr))
```