Informatics 41 • Fall 2010 • David G. Kay • UC Irvine • Your name _____

FIFTH QUIZ

You have 15 minutes from the start of class to complete this quiz. Read the problems with care; work with deliberate speed. Don't give us more than we ask for. The usual instructions apply. Good luck!

You may show lists in any of three ways: `(list 73 15)`, `'(73 15)`, or `(cons 73 (cons 15 empty))`.

Problem 1 (4 points)

Evaluate each of the following expressions, remembering which functions return lists and which return single elements of lists. Use this definition independently for each of the four parts:

```
(define L (cons "pumpkin" (cons "skeleton" (cons "skull" (cons "ghost" empty)))))
```

(a) `(first L)`

(b) `(rest L)`

(c) `(empty? L)`

(d) `(first (rest L))`

(e) `(rest (rest (rest L)))`

Problem 2 (4 points)

We define a song as a structure `(define-struct song (title artist year length))` where `title` and `artist` are strings and `year` and `length` are numbers (representing the year the song was recorded and the length in seconds of the recording).

Complete the definition of the function below according to the contract and purpose statement shown. [Hint: It only takes one line of Scheme code.]

```
;; song-longer?:  song  song  ->  boolean
;; Return true if the first input song is longer than the second.

(define song-longer?
   (lambda (S1 S2)
```

Problem 3 (2 points)

In class we discussed circuit-switched networks and packet-switched networks.

(a) One describes message routing on the internet; the other describes conventional telephone service. Which is which?

(b) A transmission typically gets split up into pieces that may take different routes to the destination. Does this describe circuit-switching or packet-switching?

Problem 4 (10 points)

(a) (3 points) Complete the definition of the function below according to the contract and purpose statement shown, with one function or parameter name in each blank. The predefined function `flip-vertical` takes an image as input and returns that image flipped vertically; this is the only image function you need here.

```
;; flip-all:  list-of-images  ->  list-of-images
;; Return the input list with each image flipped vertically.
(define flip-all
  (lambda (L)
    (cond
      ((_______________ L) _______________)


      (else (_______________ (_______________ (_______________ L))

                            (_______________ (_______________ L)))))))
```

(b) (1 point) Does the function above do filtering, mapping, or folding/reducing?

(c) (4 points) Complete the definition of the function below according to the contract and purpose statement shown, with one function or parameter name in each blank. The predefined function `zero?` returns true if its argument is zero. [Hint: Parts of this function are very similar to the function above.]

```
;; remove-zeroes:  list-of-numbers  ->  list-of-numbers
;; Return the input list with any zero values removed.
(define remove-zeroes
  (lambda (L)
    (cond
      ((_______________ L) _______________)


      ((zero? (_______________ L)) (_______________ (_______________ L)))


      (else (_______________ (_______________ L) (_______________ (_______________ L)))))))
```

(d) (1 point) Does the function above do filtering, mapping, or folding/reducing?

(e) (1 point) We're taught in school not to define something in terms of itself, because that lands us in a circular argument with no way out. How do we avoid that problem when we define recursive functions (which call themselves—that's what recursive means)? In other words, what does a correctly defined recursive function have that saves it from getting stuck? [Two words are enough if you recall the proper term; in any case, you should only need a few words.]