Informatics 41 • Fall 2008 • David G. Kay & Rich Pattis • UC Irvine • Your name ____________________

EIGHTH QUIZ

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

Problem 1 (13 points)

Suppose we define a vector as follows:

```
(define V (build-vector 6 (lambda (i) (* 2 i))))
```

(a) (2 points) Draw a six-cell box showing the contents of `V` and the element number (i.e., the index or the subscript) of each element.

(b) (1 point) What is the value of `(vector-ref V 3)`?

(c) (1 point) Change your drawing above to show what happens when Scheme evaluates this expression:
`(vector-set! V 4 (* 10 (vector-length V)))`

(d) The function below performs on a vector the same task as a list-processing function you know well.

```
;; vector-do-something:  (X -> boolean)  vector-of-X  ->  vector-of-x
(define vector-do-something
  (lambda (f V)
    (local
       ((define chosen
            (filter f (build-list (vector-length V) (lambda (i) (vector-ref V i))))))
       (build-vector (length chosen) (lambda (i) (list-ref chosen i))))))
```

(d.1) (1 point) What would be a better name for this function than `vector-do-something`?

(d.2) (2 points) Write a brief purpose statement that describes what this function does.

(d.3) (2 points) Write the function `vector->list` that converts a vector to a list. You can do it by copying some of the code above.

```
;; vector->list:  vector-of-X  -> list-of-x
(define vector->list
  (lambda (V)
```

(d.4) (2 points) Write the function `list->vector` that converts a vector to a list. You can do it by copying some of the code above (and changing one name).

```
;; list->vector:  list-of-X  -> vector-of-x
(define list->vector
  (lambda (L)
```

(d.5) (2 points) Why is the vector-based version of the function of part **(d)** so much more complicated than the list-based version? In other words, what is it about vectors that makes manipulating them in this way so hard?

Problem 2 (4 points)

For each of the algorithms or operations described below, check the box corresponding most closely to its complexity (i.e., its O-notation) in the average case.

(a) In a (balanced) binary search tree of *n* restaurants, ordered by the restaurant's name, adding a new restaurant:

□ Constant—O(1) □ Logarithmic—O(log *n*) □ Linear—O(*n*) □ Quadratic—O(n^2)

(b) In a (balanced) binary search tree of *n* restaurants, ordered by the restaurant's name, finding a restaurant in the tree, given the restaurant's name:

□ Constant—O(1) □ Logarithmic—O(log *n*) □ Linear—O(*n*) □ Quadratic—O(n^2)

(c) Adding every fifth element of an *n*-element vector:

□ Constant—O(1) □ Logarithmic—O(log *n*) □ Linear—O(*n*) □ Quadratic—O(n^2)

(d) In a (balanced) binary search tree of *n* restaurants, ordered by the restaurant's name, finding all the restaurants that serve a specified dish:

□ Constant—O(1) □ Logarithmic—O(log *n*) □ Linear—O(*n*) □ Quadratic—O(n^2)

Problem 3 (3 points)

On the Deus X machine, one machine word consists of 4 bytes, or 32 bits.

(a) What's the largest number you can represent in one machine word on the Deus X, if you represent the number as ASCII characters?

(b) What's the largest number you can represent in one machine word on the Deus X, if you represent the number using binary-coded decimal (BCD)?

(c) What's the largest number you can represent in one machine word on the Deus X, if you represent the number as a binary number? The actual arithmetic for this may take you too long to do on a quiz, so just circle ***(i)***, ***(ii)***, or ***(iii)***: ***(i)*** over 100,000,000; ***(ii)*** over 65,535 but under 100,000,000; ***(iii)*** 65,535.