Informatics 41 • Fall 2005 • David G. Kay • UC Irvine Your name _____

FIFTH QUIZ

Your student ID _____

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 (5 points)

Suppose we store a collection of restaurants in a binary search tree ordered by the restaurant's name. We define nodes nodes in the tree by `(define-struct node (value left right))`, where value is a rrant structure (`(define-struct rrant (name cuisine phone dish price))`). Complete the definition of `sum-of-prices` below. All the parentheses are in the correct places and each blank should be filled by exactly one symbol, function name, or constant.

```
;; sum-of-prices: binary-search-tree -> number
;; Return the sum of the prices of all the rrants in the tree
(define sum-of-prices
    (lambda (BST)
        (cond
            ((_______________ BST) 0)

            (else (_______________

                    (_______________ (_______________ BST))

                    (_______________ (_______________ BST))

                    (_______________ (_______________ BST)))))))
```

Problem 2 (20 points)

Your electronic cookbook contains a list of recipes as defined in last week's homework:
`(define-struct recipe (title ingredients steps))`

where the title is a symbol, ingredients is a list of symbols, and steps is a list of steps (where each step is a list of symbols); for example:

```
(make-recipe 'ThaiIcedCoffee
    '(coffee sugar condensed-milk ice)
    '((brew coffee) (add sugar and condensed-milk) (pour coffee mixture over ice)))
```

You decide you need to prepare healthier food, so you make up a list of changes (e.g., butter to olive oil or sour cream to yogurt). You represent each change as a structure
`(define-struct change (old new))`

where old and new are symbols, and you create a list of changes; for example:
`(list (make-change 'butter 'olive-oil) (make-change 'sour-cream 'yogurt))`

You want to make all of these changes throughout your cookbook, changing each occurrence of an old ingredient to its corresponding new ingredient, in the ingredients list and each of the steps of each recipe in the cookbook. Thus, you will define the function `change-cookbook` that takes a list of recipes and a list of changes and returns a list of changed recipes (but before you start writing code, read all of the following advice).

- Use these four helper functions in your code. For full credit, you must provide correct definitions of *any two* of these four functions (but you should use all of them in your code even if you don't finish defining them).

```
;; make-changes-in-one-recipe: recipe (listof change) -> recipe
;; Apply each change on the list to this recipe.

;; make-one-change-in-recipe: recipe change -> recipe
;; Return recipe with the specified change made to ingredients and steps

;; make-one-change-in-steps:  (listof steps) change -> (listof steps)
;; Make the change in each of the steps on the list

;; subst:  symbol  symbol  (listof symbol)  ->  (listof symbol)
;; Change the first symbol to the second symbol wherever it occurs in the list.
```

- We do not expect you to use `map` or `filter` in your code on this quiz, but you may use them if you're confident enough to let your score depend on it.

```
;; change-cookbook: (listof recipe)  (listof change)  ->  (listof recipe)
;; Return the input list of recipes, with each recipe modified by applying each change
;; on the list of changes to the ingredients and steps of the recipe.
```