SECOND 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!

Problem 1 (4 points)

Evaluate each of the following expressions. The function even? is predefined in DrScheme; it returns true if its argument is an even number.

(a) (define income-category
    (lambda (income)
        (cond
            ((<= income 24000) 'poor)
            ((<= income 41000) 'lower)
            ((<= income 62500) 'middle)
            ((<= income 94000) 'upper)
            (else 'high))))
    (income-category 60000)

(b) (or (= 6 (* 3 4)) (even? (* 4 3))]

(c) (and (even? (/ 100 5)) (not (= 17 (/ 34 3)))]

(d) (string=? "bagel"
    (cond
        ((even? (/ 34 2)) "donut")
        ((symbol=? 'six 'VI) "tortilla")
        (else "bagel")))

Problem 2 (21 points)

(a) (2 points) Write the Scheme code to complete this definition of a structure that a bookstore might use to represent a book:

;; A book is (make-book string string number number number),
;; where the strings represent the title and the author (in that order) and
;; the numbers represent (in order) the book’s price, the number in stock,
;; and the number sold.

(b) (2 points) Write an expression that constructs and returns Matthias Felleisen’s book How to Design Programs, which costs $79.50; the store has sold 25 and has 7 in stock.

(c) (1 point) If B is a book structure, write an expression (not a whole function) that returns the price of B.
(d) (4 points) Complete the definition of this function according to the contract and purpose given. Examples and tests are not required for credit on this quiz, but it’s still a good idea to think about them.

;;; book-income: book -> number
;;; Given a book, return the total amount of money made from sales of the book
(define book-income
  (lambda (B)
    (* (book-price B) (book-sold B))))

Scoring: 1 point for attempt to multiply (any) two values
1 point for applying either book-price or book-sold to B at least once
2 points for the rest correct (partial credit okay)

(e) (5 points) Complete the definition of this function according to the contract and purpose given. (Hint: (and X Y Z), where X, Y, and Z are boolean expressions, returns true if and only if all three are true.)

;;; Return true if the two books have the same author, title, and price, and false otherwise.
(define book-same?
  (lambda (B1 B2)

Scoring: 1 point for an attempt to compare either titles, authors, or prices (equal? is OK)
1 point for using two calls to book-title, two calls to book-author, two to book-price
1 point for attempt to combine results of three comparisons (two nested ands is OK)
1 point for correct use and placement of arguments B1 and B2
1 point for the rest being correct (partial credit okay)

(f) (7 points) Complete the definition of this function according to the contract and purpose given. Where appropriate, use the functions you defined above.

;;; If the two books are the same, return a (newly created) book combining the books’ figures. If not, just return the first argument unchanged.
(define book-combine
  (lambda (B1 B2)
    (cond
      ((book-same? B1 B2)
                  (+ (book-sold B1) (book-sold B2))))
      (else B1))))

Scoring: 1 point for use of book-same? with arguments
1 point for correct cond with book-same? and return of B1 when not same.
1 point for attempt to return book structure (make-book) when books are same
1 point for calling make-book with 5 arguments (2 strings, 3 numbers)
1 point for correctly adding either in-stock fields or sold fields for B1 and B2
1 point fully correct call to make-book
1 point for everything else correct