Quiz 8b

1. (4 points)

a. (2 points) Draw the environment diagram and find what the last statement evaluates to.

(define (foo f x)

(lambda (y) (f x y)))

(define fisher (lambda (y x) (/ y x)))

(define bar (foo fisher 4))

> (bar 8)

\_\_\_\_\_

b. (2 points) Draw the environment diagram and find what the last statement evaluates to.

> (define (foo)

(define y (cons 1 2))

(lambda (y z) (set! x (cons y z))))

> (define x 1)

> ((foo) x 3)

> x

\_\_\_\_

1. (3 points) Write a procedure mr-worm that returns a “worm” procedure. Whenever this “worm” procedure is called, it appends a 1 to a list inside the procedure. This list is initially 1 element long. Whenever mr-worm procedure is called again another worm with the same length as the most recent worm is created.

> (define worm1 (mr-worm))

worm1

> (worm1)

(1 1)

> (worm1)

(1 1 1)

> (define worm2 (mr-worm))

worm2

> (worm2)

(1 1 1 1)

> (worm1)

(1 1 1 1)

1. (3 points) To get a better understanding of how classes work under the line, we will write the equivalent of a class **without using define-class**.

We give you a book class. You can assume it works like a normal class. It has two methods title and contents which returns the title and contents respectively.

> (define book1 (instantiate book ‘(CS61as is awesome) ‘(A very short book) ))

book1

> (ask book1 ‘title)

(CS61as is awesome)

> (ask book1 ‘contents)

(A very short book)

**Now write code that will simulate a shelve class with a list of books.** It should be able to add books when given a book. It should output false when finding a book the shelf doesn’t have. Assume there will be no books with the same title. Remember write this in below the line. You can assume book works above the line though.

> (define shelf1 (make-shelf)

shelf1

> ((shelf1 ‘find) ‘(The Meaning Of Life))

#f

> ((shelf1 ‘add) book1)

okay

> (ask ((shelf1 ‘find) ‘(CS61as is awesome)) ‘contents)

(A very short book)

Here is the above line version:

(define-class (shelf)

(instance-vars (books nil))

(method (add-book title) #code to add a book to books#)

(method (find-book title) #code to find a book in books#))