Quiz 2b

1. (1 point each) Write down what each expression returns. If it errors, write ‘error’. If it returns a procedure, write ‘proc’



>(define (proc x y)

(let ((y ‘foo))

(word y x)))

>(proc ‘bar ‘lo)

\_\_\_’**foobar**\_\_\_\_\_\_\_

All or nothing

1. (2 points) Define mystery so that the following line outputs 9.

> ((mystery 3) 3)

-2 points: evaluates to 9

-1 point: (mystery 3) returns a function that accepts one argument

1. (3 points) Write the function my-keep that takes in a predicate and a list **or** a word and outputs a sentence whose elements are elements from the input that evaluates the predicate to true.

>(my-keep even? ‘(1 2 3 4))

(2 4)

>(my-keep vowel? ‘foobar)

(o o a)

(define (my-keep pred input)

   (cond ((empty? input) ‘())

             ((pred (first input)) (se (first input) (my-keep pred (bf input))))

             (else (my-keep pred (bf input))))))

-3 points: correct solution

-2 points: only missing base case

OR only works on lists

OR only works on sentences

-1 point: Correct structure (checks with pred, correct recursive calls, but not functional)

1. (3 points) Write a function do-n-times that takes in a function and a nonnegative number, which returns a procedure that performs the function n times

> ((do-n-times bf 3) ‘hello)

lo

> ((do-n-times (lambda (x) (\* x 2)) 3) 1)

8

>((do-n-times (lambda (wd) (word wd ‘s)) 4) ‘hi)

Hissss

(define (do-n-times func n)

   (lambda (x)

       (if (= n 1)

           (func x)

           (func ((do-n-times func (- n 1)) x)))))

-3 points: correct.

-2 points: do-n returns a function , but does not do the function n times

-1 point: do-n returns a function, but not recursive/structure is wrong