Quiz 12a

1. (2 points) Consider the following Scheme code for the **lazy** evaluator:

(define counter 0)

(define (increment!)

(set! counter (+ counter 1)) counter)

(define (count-down n val)

(if (= n 0)

val

(count-down (- n 1) (increment!))))

What is the return value of (count-down 3 0)?

1, 2 points are all or nothing.

1. (3 points) In the **lazy** evaluator we do this:

> (define w 100)

> (define (foo x y)

(x y))

> (define q (foo (lambda (z)

(set! w 50)

z)

(begin (set! w 10)

3)))

> w

\_\_\_\_\_\_50\_\_\_\_\_\_

> q

\_\_\_\_\_\_\_3\_\_\_\_\_

> w

\_\_\_\_\_\_\_10\_\_\_\_\_

Fill in the blanks with the correct values.

One point each all or nothing.

1. (2 points) True or false. **Justify your answer.**
   1. **true false** When using the lazy evaluator, a procedure call can be delayed.

**True**, all arguments to compound procedures are delayed that includes procedure calls.

* 1. **true false** If cons, car, and cdr are provided as primitives in the lazy evaluator, all lists will automatically be streams.

**False**, primitives force arguments so the cons will force the cdr.

Half point for correct answer and half point for justification on each part.

1. (3 points) Given definitions:

> (define (foo x y)

(if (> x y)

(/ x y)

x))

> (define (bar x y)

(if (> x 1)

(/ x y)

x))

For each of the following interactions, circle TRUE if the **lazy** evaluator is more efficient than the regular metacircular evaluator, or FALSE if it isn't any faster.

* 1. > (foo (+ 2 3) (\* 5 2))

TRUE ***FALSE***

* 1. > (bar (foo 10 5) 1)

TRUE ***FALSE***

* 1. > (bar 1 (foo 10 5))

***TRUE*** FALSE

One point each all or nada.