Assignment 7

Programming Languages

1. [20 pts] Convert the following Scheme code to tail recursive:

```
(define length
  (lambda (lat)
    (cond
      ((null? lat) 0)
      (else (+ 1 (length (cdr lat))))
    )))

(length '(a b c d e f))
=> 6
```

2. [40 pts] Write a Scheme function `(mean lst)` to compute the mean value of a list of integers. Make sure the function traverses the list once and computes both the sum and the length in order to return the mean (i.e. sum/len).

3. [40 pts] Recall that lazy variant or call-by-need, is an evaluation strategy which delays the evaluation of an expression until its value is needed and which also avoids repeated evaluations. Given,

```
(define foo
  (lambda (x y z)
    (if x (1+ y) (1- z))))
```

(a) What happens if we enter

```
(foo #t 7 (quotient 1 0))
```

i.) in Scheme?

ii.) in a lazy variant of Scheme?

(b) What happens if we enter

```
(foo #t 2 (letrec ([loopy (lambda (x)
                            (if (zero? x)
                                0
                                (loopy (1+ x))))])
            (loopy 1)))
```

i.) in Scheme?

ii.) in a lazy variant of Scheme?