Jacob Petrisko

Q1.) what does this return?

```
(cons '(PLC) (append '(ISEASY) (call/cc (lambda (c) (* 3 a (a (list? a)))))))
```

r_1: (λ (a) (* 3 a (a (list ? a))))

k_1: (escaper (λ (v) (cons '(PLC) (append '(ISEASY) v))))

(r_1 k_1)

=> (λ (k_1) (* 3 k_1 (k_1 (list ? k_1))))

(list ? (escaper λ (v) (cons '(PLC) (append '(ISEASY) v))))

#f escaper is a procedure

escape

=> (cons '(PLC) (append '(ISEASY) #f))

=> ((PLC) ISEASY . #f)

Q2.) Write factorial in scheme, then write (in scheme code) how you would execute it using an engine.

```
(define factorial
  (λ (n)
    (cond ((< n 0) #f)
          ((<= n 1) 1)
          (else (* n (factorial (- n 1)))))))

(define engine-fact
  (λ (n)
    (make-engine
      (λ () (factorial n)))))

(define eng (engine-fact 5))
(eng 150 cons (λ (new-eng) (set! eng new-eng)))
```