CSSE 304 Quiz Day 32 points Name _____ **Section** 01 (9:00) 02 (9:55) 03 (10:50)

1. We know that **(call/cc receiver)** is equivalent to **(receiver continuation)**.

(9) Put an X in the appropriate box for each row.

Procedure	Never an escape procedure	Can be an escape procedure, but is not always an escape procedure	Always an escape procedure
call/cc	X		
receiver		X	
continuation			X

2. Consider the code below. You do not have to write the details if you can answer the questions without doing so. But you may find that doing so helps you to get the right answers.

(a) (6) What does it print? _____ 112 _____

(b) (10) How many times is **call/cc** applied during the execution of the code? ___ 5 _____

```
(define s
    (lambda (x)
      (display 1)
      (call/cc (call/cc x))
      (display 2)))
(s (call/cc (lambda (v) v)))
```

This is the "full text" narration. If I was just figuring it out and not explaining it, I would have probably only write ¼ as much.

As in the `strange1` example from class, for the first application of call/cc, r1 = identity, k1 = `(escaper s)`.
So `(s (call/cc (lambda (v) v)))` becomes `(s ((lambda (v) v) k1))`, same as `(s k1)`. ❶
Thus we execute the body of `s` with x = k1.

`(begin (display 1) (call/cc (call/cc k1)) (display 2))`. It displays 1, and then we have
`(begin (call/cc (call/cc k1)) (display 2))` The next thing to execute is the inner call/cc application.

r2 = k1, k2 = `(escaper (lambda (v) (call/cc v) (display 2)))`. So the above becomes ❷
`(begin (call/cc (k1 k2)) (display 2))`. k1 is an escape procedure; this becomes `(k1 k2)`, same as `(s k2)`.

`(begin (display 1) (call/cc (call/cc k2)) (display 2))`. It displays 1, and then we have
`(begin (call/cc (call/cc k2)) (display 2))` The next thing to execute is the inner call/cc application.

r3 = k2, k3 = `(escaper (lambda (v) (call/cc v) (display 2)))`. So the above becomes ❸
`(begin (call/cc (k2 k3)) (display 2))`. k2 is an escape procedure; this becomes `(k2 k3)`
Thus we execute the body of k2, with v = k3. This gives us `(begin (call/cc k3) (display 2))`

r4 = k3, k4 = `(escaper (lambda (v) v (display 2)))`. Above code becomes `(begin (k3 k4) (display 2))`. ❹
k3 is an escape procedure; this becomes `(k3 k4)`, Thus we execute the body of k3, with v = k4. This is
`(begin (call/cc k4) (display 2))`.

r5 = k4, k5 = `(escaper (lambda (v) v (display 2)))`. Above code becomes `(begin (k4 k5) (display 2))`. ❺
k4 is an escape procedure; this becomes `(k4 k5)`, Thus we execute the body of k4, with v = k5. k4 is the constant procedure that (no matter what its argument is) displays 2. Nothing is returned, so nothing else gets printed.