CSSE 304 Days 29-30 Summary `call/cc`:

1. **call/cc definition**
   a. **call/cc** is an abbreviation for **call-with-current-continuation**
   b. call/cc is a procedure that takes one argument; the argument is a *receiver*.
   c. This receiver is a procedure that takes one argument; that argument (in this case) is a *continuation.*
   d. A continuation is a procedure (that takes one argument); that continuation embodies the context of the application of **call/cc**. It is an escape procedure.
   e. The application **(call/cc receiver)** has the same effect as **(receiver continuation)**, where the continuation is
   f. an escape procedure that embodies the execution context of the entire **call/cc** expression.
2. **call/cc definition summary**:
   a. **(call/cc receiver)** ➔ (**receiver continuation)**,
   b. Hence the name: call-with-current-continuation.
   c. **Rephrasing it:** What is that continuation?
      If **c** is a procedure that represents the execution context of this application of **call/cc**, then the continuation is equivalent to **(escaper c).**
3. **call/cc example**
   a. **(call/cc receiver)** ➔**(receiver continuation)**,
   b. Consider `(+ 3 (call/cc (lambda (k) (* 2 (k 5)))))`
   c. The receiver is (the closure created by executing)
   d. The context c is
   e. The continuation is
   f. Thus `(+ 3 (call/cc (lambda (k) (* 2 (k 5)))))` is equivalent to

4. More `call/cc` examples

```
(+ 3 (call/cc (lambda (k) (* 2 5))))
```

```
(+ 3 (call/cc (lambda (k)   (k (* 2 5)))))
```

5.
```
(define xxx #f)
(+ 5 (call/cc (lambda (k)
                (set! xxx k)
                2))) ; xxx is equivalent to?
 (* 7 (xxx 4))
```

6.
```
(call/cc procedure?)
```

7. `list-index` example is detailed in the slides:

8. **((car (call/cc list)) (list cdr 1 2 3))**

9.
```
(define strange1
  (lambda (x)
    (display 1)
    (call/cc x)
    (display 2)))

(strange1 (call/cc (lambda (k) k)))
```

(define strange2 ; try this one yourself soon
(lambda (x)
(display 1)
(call/cc (lambda (j) (x j)))
(display 2)
(call/cc (lambda (c) (x c)))
(display 3)))

(strange2 (call/cc (lambda (k) k)))