8-1

Handler — ISWIM

$M = \ldots$
$\quad | \ \text{throw } M$
$\quad | \ \text{catch } M_1 \text{ with } \lambda X. M_2$

loop :=

```
catch
    if input is bad
       throw "Bad User"
       f(input)
with
 λX. loop()
```

$E[b\ V] \mapsto E[\text{throw errnotafun}]$

$E = [\,] \ | \ (E\ M) \ | \ (V\ E) \ | \ (o^n\ V\ldots\ E\ M\ldots)$
$\quad | \ \text{throw } E$
$\quad | \ \text{catch } E \text{ with } \lambda X. M$

$E[\text{catch } V \text{ with } \lambda X. M] \mapsto E[V]$

(catch (+ (catch (+ 5 (throw 6)) with λX.X) 1) with λY.Y) $\mapsto^* 6$

$E[\text{throw } 6] \mapsto E'[(\lambda X.X)\ 6]$

$E = $ (catch (+ (catch (+ 5 [ ]) with λX.X) 1) with λY.Y) $E' = $ (catch (+ [ ] 1) with λY.Y)

$E[\text{throw } V] \mapsto E'[(\lambda X.M)\ V]$
$(E', \lambda X.M) = \mathcal{F}(E)$
$\mathcal{F}(\text{catch } E \text{ with } \lambda X.M) = (E, \cancel{\lambda X.M})$
$\mathcal{F}(E)$ if it returns ...
o/w, $([\,], \lambda X.M)$

F is "catch"-less evaluation context

$F = \blacksquare \mid (V\ F) \mid (F\ M) \mid (o^n\ V \ldots\ F\ M \ldots) \mid (\text{throw}\ F)$

$E = \blacksquare \mid (V\ E) \mid (E\ M) \mid (o^n\ V \ldots\ E\ M \ldots) \mid (\text{throw}\ E)$
$\quad \mid \text{catch}\ E\ \text{with}\ (\lambda X.M)$

$E[(\lambda X.M)\ V] \mapsto E[M[X \leftarrow V]]$
$E[(o^n\ V \ldots)] \mapsto E[\delta(o^n, V \ldots)]$
$E[b\ V] \mapsto E[\text{throw errnaf}]$
$E[\text{catch}\ V\ \text{with}\ \lambda X.M] \mapsto E[V]$

$E[\text{catch}\ F[\text{throw}\ V]\ \text{with}\ \lambda X.M]$
$\mapsto E[(\lambda X.M)\ V]$ F isn't on the RHS

~~$\text{eval}(M) = b$ if $M \mapsto^* b$~~
~~'fun if $M \mapsto^* (\lambda X.M)$~~

eval(throw 6) is stuck

$\text{eval}(M) = b$ if $M' \mapsto^* b$
'fun if $M' \mapsto^* (\lambda X.M)$
$M' = \text{catch}\ M\ \text{with}\ (\lambda X.X)$

Handle-CEK $K = \ldots \mid \text{mt} \mid \text{fun}(M,E,K)$
$\mid \text{arg}(V,K)$
$\mid \text{throw}(K)$
$\mid \text{catch}(\lambda X.M, E, K)$

$\langle \text{throw}\ M, E, K \rangle \mapsto \langle M, E, \text{throw}(K) \rangle$

$\langle \text{catch}\ M_1\ \text{with}\ \lambda X.M_2, E, K \rangle \mapsto \langle M_1, E, \text{catch}(\lambda X.M, E, K) \rangle$

$\langle V, E, \text{catch}(\lambda X.M, E, K) \rangle \mapsto \langle V, E, K \rangle$

$\langle V, E, \text{throw}(\text{throw}(K)) \rangle \mapsto \langle V, E, \text{throw}(K) \rangle$

$\langle V, E, \text{throw}(\text{fun}(M, E', K)) \rangle \mapsto \langle V, E, \text{throw}(K) \rangle$
$\langle V, E, \text{throw}(\text{catch}(\lambda X.M, E', K)) \rangle \mapsto \langle ((\lambda X.M)\ V), E', K \rangle$

1-3/

Recoverable - ISWIM

$M = \ldots$
$\quad |\ \text{throw } M$
$\quad |\ \text{catch } M_1 \text{ with } \overset{\text{exception (ie thrown) value}}{\lambda X.}\overset{\text{recover function}}{\lambda R.}\ M_2$

(+ (catch (+ 5 (throw 6))
    with (λX.λR. (R (+ X X))))
  2) $\longrightarrow^*$ 19 $\hookrightarrow$ λY. (+ 5 Y)

F / E difference in eval-ctx

① E[catch F[throw V] with (λX.λR. M)]
$\quad \mapsto$ E[(λX.λR. M) V $\underline{\quad R_V \quad}$] $\qquad = 6$
$\qquad R_V = \lambda Y.\ F[Y]$

(+ (catch (+(throw 5) (throw 6))
    with (λX.λR. (R (+ X X))))
  2) $\longrightarrow^*$ 24? 6?

② E[catch F[throw V] with (λX.λR. M)]
$\quad \mapsto$ E[~~catch (λX.~~ catch ((λX.λR. M) V (λY. F[Y])))]
$\qquad\qquad\qquad\qquad$ with (λX.λR. M)

③ $\mapsto$ E[((λX.λR. M) V (λY. catch F[Y] with (λX.λR. M))]

(catch (+ 1 (let k := throw 0 in
$\qquad\qquad\quad$ (+ (k 1) (k 1))))
  with (λX.λR. (R R)))
$\qquad\qquad\quad \hookrightarrow$ (λY. (+ 1 Y))
$\qquad\qquad\quad \hookrightarrow$ (λY. (+1 (let k := Y
$\qquad\qquad\qquad\qquad$ (+ (k 1) (k 1))))

18-4/ Full Control - ISWIM

```
M = ...                E = ...
  | (call/cc M)          | (call/cc E)
```

$$\boxed{E[(\text{call/cc } V)] \mapsto E[V\ (\lambda Y.\ E[Y])]}$$

```
(define threads empty)
(define (yield)
  (call/cc r
    (λ (resume-me)
      (~~kernel reso~~ (set! threads (~~cons~~ snoc threads resum-me
      (schedule!))))
(define (schedule!)
  (define next (first threads))
  (set! threads (rest threads))
  (next "Your turn again"))
```

```
(define last-handler default)
(define (throw v) (last-handler v))
(define (catch body H)
(call/cc (λ (resume)
  (define old-H last)
  (set! last (λ (X) (set! last old)
               (resume (H X))))

  (body) )

              λX. call/cc (λ recover.
                 (resume (H X recover))
```

$\langle V, E, \text{call/cc}(k) \rangle \mapsto \langle (V\ k), E, k \rangle$

$\langle (k\ V), E, k' \rangle \mapsto \langle V, E, k \rangle$