1. Question 1: Exploring m-eval

```
;question 1
(list '* *)
(list '/ /)
(list 'list list)
(list 'cadr cadr)
(list 'cddr cddr)
(list 'newline newline)
(list 'printf printf)
(list 'length length)

;Question 1: Exploring m-eval

;test-case
(* 5 10)
(/ 5 10)
(list 1 2 3)
(define x (list 1 2 3 4 5))
(cadr x)
(cddr x)
(printf "print")
(newline)
(length x)
```

primitive-procedures에 *, /, list, cadr, cddr, newline, printf, length 등을 추가하고 test-case로 실행하기

1-1. test-case

```
;;; M-Eval input level 1
(* 5 10)

;;; M-Eval value:
50
```

```
;;; M-Eval input level 1
(/ 5 10)

;;; M-Eval value:
1/2
```

```
;;; M-Eval input level 1
(printf "print")
print
;;; M-Eval value:
#<void>
```

```
;;; M-Eval input level 1
(newline)

;;; M-Eval value:
#<void>
```

```
;;; M-Eval input level 1
(list 1 2 3)

;;; M-Eval value:
(1 2 3)
```

```
;;; M-Eval input level 1
(define x (list 1 2 3 4 5))

;;; M-Eval value:
#<void>
```

```
;;; M-Eval input level 1
(length x)

;;; M-Eval value:
5
```

```
;;; M-Eval input level 1
(cadr x)

;;; M-Eval value:
2
```

```
;;; M-Eval input level 1
(cddr x)

;;; M-Eval value:
(3 4 5)
```

2. Question 2: Adding a special form directly

```
;Question 2: Adding a special form directly

(define (and? exp) (tagged-list? exp 'and))
(define (and-val exp) (cdr exp))
(define (and-first exp) (car (and-val exp)))
(define (and-second exp) (cdr (and-val exp)))
(define (and-empty-first? exp) (null? (and-val exp)))
(define (and-empty-second? exp) (null? (cdr (and-val exp))))
(define (make-and clauses) (cons 'and clauses))
(define (eval-and exp env)
  (cond
    ((and-empty-first? exp) #t)
    ((and-empty-second? exp) (m-eval (and-first exp) env))
    (else
      (and
        (m-eval (and-first exp) env)
        (eval-and (make-and (and-second exp)) env)))))

 ;2
 ((and? exp) (eval-and exp env))
```

eval-and, and?, and-clauses 구현, m-eval에 and? 추가.

2-1. test-case

```
;;; M-Eval input level 1
(and)

;;; M-Eval value:
#t
```

```
;;; M-Eval input level 1
(and 5)

;;; M-Eval value:
5
```

```
;;; M-Eval input level 1
(and 1 (< 3 1) 3)

;;; M-Eval value:
#f
```

```
;;; M-Eval input level 1
(and #f)

;;; M-Eval value:
#f
```

```
;;; M-Eval input level 1
(and 1 2 3)

;;; M-Eval value:
3
```

```
;;; M-Eval input level 1
(define x 3)

;;; M-Eval value:
#<void>
```

```
;;; M-Eval input level 1
(and #f (set! x 500000))

;;; M-Eval value:
#f
```

```
;;; M-Eval input level 1
(and (set! x 500000) x)

;;; M-Eval value:
500000
```

```
;;; M-Eval input level 1
(and (< x 5) (= 6 6) (null? '()))

;;; M-Eval value:
#t
```

```
;;; M-Eval input level 1
(and x)

;;; M-Eval value:
3
```

```
;;; M-Eval input level 1
(and x)

;;; M-Eval value:
500000
```

3. Question 3: Adding a special form via transformer

```
;Question 3: Adding a special form via transformer

(define (until? exp) (tagged-list? exp 'until))
(define (until-test exp) (cadr exp))
(define (until-exps exp) (cddr exp))

 (define (until->transformed exp)
   `(let ()
     (define (loop)
       (if (until-test exp)
         #t
         (begin
           (until-exps exp)
           (loop))))
     (loop)))

    ;3
    ((until? exp) (m-eval (until->transformed exp) env))
```

until 구현 m-eval에 until? 추가

3-1. test-case

```
;test-case
 (define u1 '(until (> x n) (printf "~s~n" x) (set! x (+ x 1))))
 (until->transformed u1)

 (define u2 '(until test))
 (until->transformed u2)

 (define u3 '(until test exp1))
 (until->transformed u3)

 (define u4 '(until test exp1 exp2 exp3))
 (until->transformed u4)
```

```
(let ()
  (define (loop)
    (if (until-test exp) #t (begin (until-exps exp) (loop))))
  (loop))
(let ()
  (define (loop)
    (if (until-test exp) #t (begin (until-exps exp) (loop))))
  (loop))
(let ()
  (define (loop)
    (if (until-test exp) #t (begin (until-exps exp) (loop))))
  (loop))
(let ()
  (define (loop)
    (if (until-test exp) #t (begin (until-exps exp) (loop))))
  (loop))
>
```

```
;;; M-Eval input level 1
(define x 0)

;;; M-Eval value:
#<void>


;;; M-Eval input level 1
(define n 5)

;;; M-Eval value:
#<void>


;;; M-Eval input level 1
(until (> x n) (printf "~s~n" x) (set! x (+ x 1)))
0
1
2
3
4
5

;;; M-Eval value:
#t
```

4. Question 4: Undoing assignments

```
;Question 4: Undoing assignments

(define (one-binding-value? binding) (null? (cdddr binding)))
(define (set-binding-value! binding val)
  (if (binding? binding)
      (set-cdr! (cdr binding) (cons val (cddr binding)))
      (error "Not a binding: " binding)))

(define (unset-binding-value! binding)
  (cond
    ((not (binding? binding)) (error "Not a binding: " binding))
    ((one-binding-value? binding) (void))
    (else
      (set-cdr! (cdr binding) (cdddr binding)))))

(define (reset-binding! binding val)
  (if (binding? binding)
      (set-cdr! (cdr binding) (cons val '()))
      (error "Not a binding: " binding)))

(define (unset? exp) (tagged-list? exp 'unset!))
(define (unset-variable exp) (cadr exp))
(define (eval-unset exp env)
  (let ((var (unset-variable exp)))
    (let ((binding (find-in-environment var env)))
      (if binding
        (unset-binding-value! binding)
        (error "Unbound variable -- UNSET" var))))

;4
((unset? exp) (eval-unset exp env))
```

```
; (define (set-binding-value! binding val)
;   (if (binding? binding)
;       (set-car! (cddr binding) val)
;       (error "Not a binding: " binding)))
```

```
(define (define-variable! var val env)
  (let ((frame (environment-first-frame env)))
    (let ((binding (find-in-frame var frame)))
      (if binding
          (reset-binding! binding val) ;4
          (add-binding-to-frame!
           (make-binding var val)
           frame)))))
```

unset! 구현 m-eval에 unset? 추가 define-variable!에서 reset-binding! 부분 변경.
set-binding-value!를 새로 디파인 했으므로 기존의 set-binding-value!는 주석처리.

4-1. test-case

```
;;; M-Eval input level 1
(define x 5)

;;; M-Eval value:
#<void>


;;; M-Eval input level 1
(set! x 10)

;;; M-Eval value:
#<void>


;;; M-Eval input level 1
(set! x 20)

;;; M-Eval value:
#<void>


;;; M-Eval input level 1
x

;;; M-Eval value:
20
```

```
;;; M-Eval input level 1
(unset! x)

;;; M-Eval value:
#<void>


;;; M-Eval input level 1
x

;;; M-Eval value:
10


;;; M-Eval input level 1
(unset! x)

;;; M-Eval value:
#<void>


;;; M-Eval input level 1
x

;;; M-Eval value:
5
```

5. Question 5: Making environments first-class

```
;Question 5: Making environments first-class

(define (boxed-env? boxed-env)
  (and
    (box? boxed-env)
    (environment? (unbox boxed-env))))
(define (box-env env)
  (if (environment? env)
    (box-immutable env)
    (error "Not an environment: " env)))
(define (unbox-env boxed-env)
  (if (boxed-env? boxed-env)
    (unbox boxed-env)
    (error "Not an environment: " boxed-env)))

(define (current-env? exp) (tagged-list? exp 'current-env))
(define (eval-current-env env) (box-env env))

(define (procedure-env? exp) (tagged-list? exp 'procedure-env))
(define (eval-procedure-env exp env)
  (box-env (procedure-environment (m-eval (second exp) env))))
```

current-env, procedure-env 구현.

```
;5
(define (env-variables boxed-env)
  (frame-variables
    (environment-first-frame
      (unbox-env boxed-env))))

(define (env-parent boxed-env)
  (box-env (enclosing-environment (unbox-env boxed-env))))

(define (env-value sym boxed-env)
  (if (symbol? sym)
    (let ((binding (find-in-environment sym (unbox-env boxed-env))))
      (if binding
        (binding-value binding)
        #f))
    (error "Not a symbol: " sym)))
;
```

```
;5
((current-env? exp) (eval-current-env env))
((procedure-env? exp) (eval-procedure-env exp env))
((application? exp)
```

primitive-procedure에 세 가지 추가.

5-1. test-case

```
;;; M-Eval input level 1
(define (make-counter) (let ((n 0)) (lambda () (set! n (+ n 1)) n)))

;;; M-Eval value:
#<void>


;;; M-Eval input level 1
(define c (make-counter))

;;; M-Eval value:
#<void>


;;; M-Eval input level 1
(c)

;;; M-Eval value:
1


;;; M-Eval input level 1
(c)

;;; M-Eval value:
2


;;; M-Eval input level 1
(env-value 'n (procedure-env c))

;;; M-Eval value:
2
```

6. Question 6: m-evals all of the way down

```
;6
(define (until->transformed exp)
  (make-let
    '()
    (list
      (make-define
        '(loop)
        (make-if
            (until-test exp)
            #t
            (make-begin (append (until-exps exp) (list '(loop))))))
      '(loop))))
```

```
;6
(list 'caddr caddr)
(list 'cadddr cadddr)
(list 'caadr caadr)
(list 'cdadr cdadr)
(list 'symbol? symbol?)
(list 'pair? pair?)
(list 'number? number?)
(list 'string? string?)
(list 'boolean? boolean?)
(list 'append append)
(list 'eq? eq?)
(list 'equal? equal?)
```

load-meval-defs, time 구현. question3 redefine, 및 primitive-procedures에 구현에 필요한 요소 추가.

6-1. test-case

```
(define (fib n) (if (< n 2) n (+ (fib (- n 1)) (fib (- n 2)))))
(time (fib 8))    ;; Should print "cpu time: 0 real time: 0 gc t
; (load-meval-defs) ;; Should print "loaded"
; (driver-loop)     ;; Should go to M-Eval input level 1
```

```
환영합니다. DrRacket, 버전 7.0 [3m].
언어: racket, with debugging; memory limit: 512 MB.
cpu time: 0 real time: 0 gc time: 0
21
>
```

enviroment 밖에서의 time.

```
loaded
> (driver-loop)


;;; M-Eval input level 1
(define (fib n) (if (< n 2) n (+ (fib (- n 1)) (fib (- n 2)))))

;;; M-Eval value:
#<void>


;;; M-Eval input level 1
(time (fib 8))
cpu time: 15 real time: 16 gc time: 0

;;; M-Eval value:
21
```

level1 에서의 time

level2 에서의 time.

```
;;; M-Eval input level 1
(driver-loop)


;;; M-Eval input level 2
(define (fib n) (if (< n 2) n (+ (fib (- n 1)) (fib (- n 2)))))

;;; M-Eval value:
#<void>


;;; M-Eval input level 2
(time (fib 8))
cpu time: 906 real time: 906 gc time: 32

;;; M-Eval value:
21
```

```
;;; M-Eval input level 2
**quit**

;;; M-Eval value:
meval-done


;;; M-Eval input level 1
(load-meval-defs)

;;; M-Eval value:
loaded


;;; M-Eval input level 1
(driver-loop)


;;; M-Eval input level 2
(driver-loop)


;;; M-Eval input level 3
```

```
;;; M-Eval input level 3
(time (+ 3 4))
cpu time: 5234 real time: 5233 gc time: 155

;;; M-Eval value:
7
```

level2에서 나온 후, load-meval-defs 실행, level3 까지 진입한 후 3 + 4의 time.

```
> (driver-loop)


;;; M-Eval input level 1
(define (fib n) (if (< n 2) n (+ (fib (- n 1)) (fib (- n 2)))))

;;; M-Eval value:
#<void>


;;; M-Eval input level 1
(time (fib 8))
cpu time: 0 real time: 0 gc time: 0

;;; M-Eval value:
21
```

level1에서 피보나치의 time.

level2에서 피보나치의 time.

```
;;; M-Eval input level 1
(driver-loop)


;;; M-Eval input level 2
(define (fib n) (if (< n 2) n (+ (fib (- n 1)) (fib (- n 2)))))

;;; M-Eval value:
#<void>


;;; M-Eval input level 2
(time (fib 8))
cpu time: 4593 real time: 4592 gc time: 32

;;; M-Eval value:
21
```

level3에서는 하루 종일 걸려서 측정하기 어려움이 있었다.