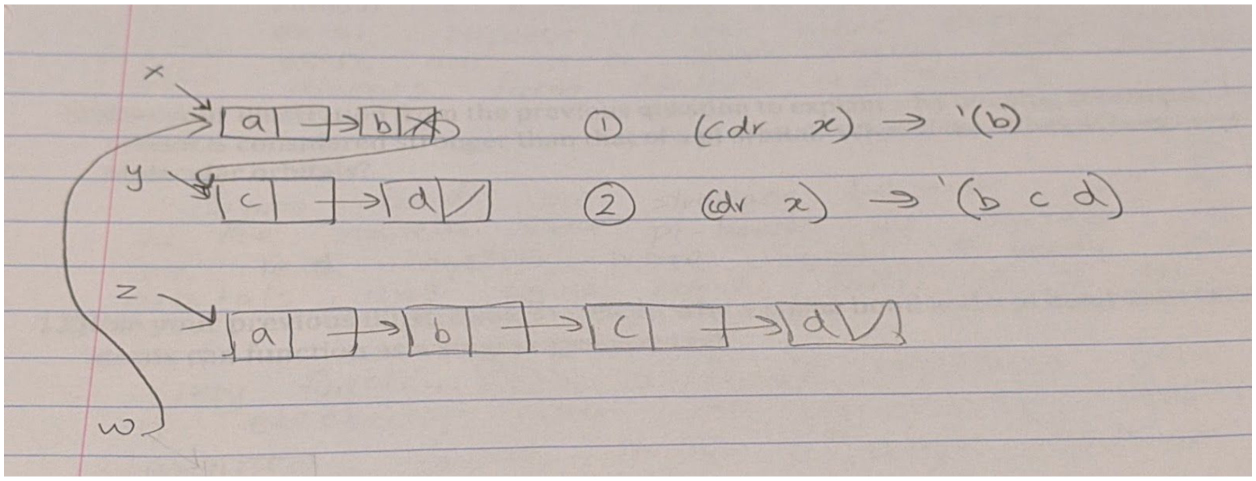
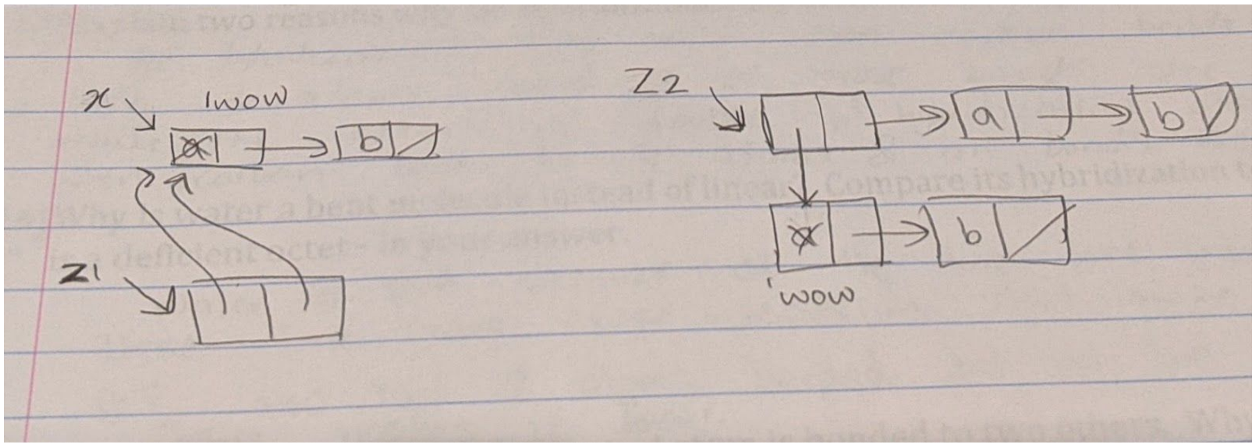
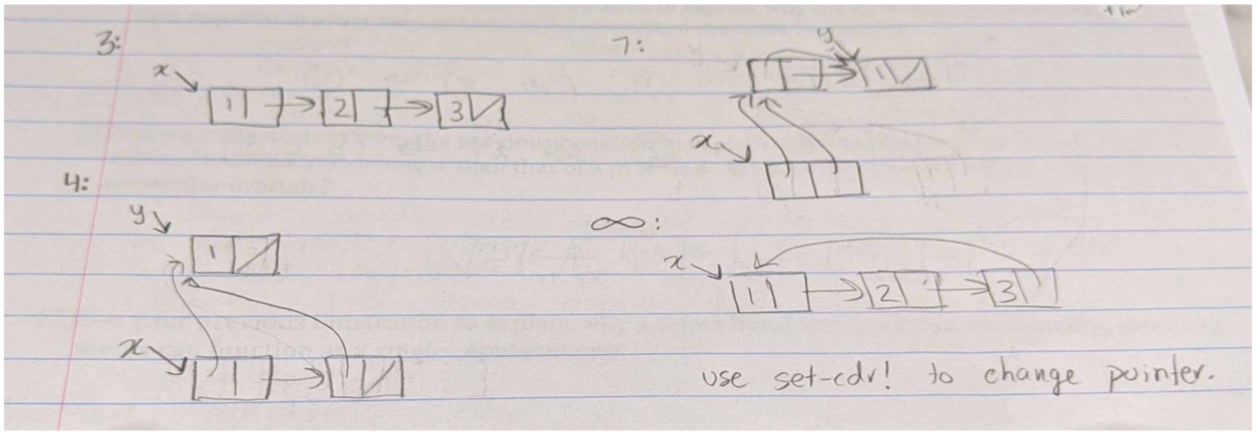
3.12


x
y
z
w
a
b
c
d
① (cdr x) → '(b)
② (cdr x) → '(b c d)

3.15


x
'wow
z1
z2
a
b
'wow

3.16


3:
x
1
2
3
4:
y
x
7:
∞:
use set-cdr! to change pointer.

3.22

```
(define (make-queue)
  (let ((front-ptr '())
        (rear-ptr '()))

    (define (front-queue)
      (if (null? front-ptr)
          'EMPTY
          (car front-ptr)))

    (define (rear-queue)
      (if (null? front-ptr)
          'EMPTY
          (car rear-ptr)))

    (define (insert-queue! item)
      (let ((new-pair (cons item '())))
        (cond ((null? front-ptr)
               (set! front-ptr new-pair)
               (set! rear-ptr new-pair))
              (else
               (set-cdr! rear-ptr new-pair)
               (set! rear-ptr new-pair)))))

    (define (delete-queue!)
      (cond ((null? front-ptr)
             '(DELETE! called with an empty queue))
            (else
             (set! front-ptr (cdr front-ptr)))))

    (define (dispatch m)
      (cond ((eq? m 'insert) insert-queue!)
            ((eq? m 'delete) (delete-queue!))
            ((eq? m 'print) (print-queue))
            ((eq? m 'front) (front-queue))
            ((eq? m 'rear) (rear-queue))
            (else
             '(Error operator))))
    dispatch))
```