The `lat?` Function

```
(define lat?
    (lambda (l)
      (cond
        ((null? l) #t)
        ((atom? (car l)) (lat? (cdr l)))
        (else #f))))
```

The `member?` Function

```
(define member?
    (lambda (a lat)
      (cond
        ((null? lat) #f)
        ((eq? a (car lat) #t)
        (else (member? a (cdr lat))))))
```

The Skeleton of a Function

```
(define function
    (lambda (parameters)
      (cond
        (q1 ans)
        (q2 ans)
          ...
        (else ans) )))
```

The `rember` Function

```
(define rember
 (lambda (a lat)
  (cond
   ((null? lat) '())
   ((eq? a (car lat)) (cdr lat))
   (else (cons (car lat)
               (rember a (cdr lat)))))))
```

The `firsts` Function

```
(define firsts
 (lambda (lat)
  (cond
   ((null? lat) ...)
   (else (cons (car (car lat))
               (firsts (cdr lat)))))))
```

The `seconds` Function

```
(define seconds
 (lambda (lat)
  (cond
   ((null? lat) ...)
   (else (cons (car (cdr (car lat)))
               (seconds (cdr lat)))))))
```

The `insertR` Function

```
(define insertR
 (lambda (n o lat)
  (cond
   ((null? lat) ...)
   ((eq? (car lat) o) (cons o
                            (cons n
                              (insertR n
                                       o
                                       (cdr l))
                            )
                       )
    )
   (else (cons (car (cdr (car lat)))
               (seconds (cdr lat))))))
```

lat?

```
(lat? '(a e sda 12 78 hi))
(lat? '(() () ()))
(lat? '())
(lat? '(883274638746328746))
(lat? '(a b c d e))
(lat? '(a aa aaa aaaa))
(lat? '(1 2 3 ()))
(lat? '((a)(b)(c)))
(lat? '(()))
(lat? '(a v c d e (a v c d e)))
```

member?

```
(member? 'z '(a e sda 12 78 hi))
(member? 'z '())
(member? 'z '(a))
(member? 'hi '(883274638746328746))
(member? 'hi '(hello hi))
(member? 'hi '(a aa aaa aaaa))
(member? 'hi '(1 2 3))
(member? 'hello '(hello))
(member? 'abc '(a b c))
(member? 'a '(a a a))
(member? 'a '(abc ade agh))
```

rember

```
(rember 'sda '(a e sda 12 78 hi))
(rember 'z '())
(rember 'a '(a))
(rember 'hi '(88327463874632 8746))
(rember 'hi '(hello hi))
(rember 'hi '(a aa hi aaa aaaa))
(rember 'hi '(hi 1 2 3))
(rember 'hello '(hello))
(rember 'abc '(a b c))
(rember 'a '(abc ade agh))
(rember 'a '(a a a))
```

firsts

```
(firsts '((a b) (e 2) (sda 12)) )
(firsts '((a b) (a 2) (a 45)) )
(firsts '() )
(firsts '(()) )
(firsts '(a) )
(firsts '((a)) )
(firsts '((a b) )
(firsts '((a b) (e 2)) )
(firsts '((a b c) (e 2 d)) )
```

seconds

```
(seconds '((a b)(e 2) (sda 12)) )
(seconds '((a b)(a 2) (a 45)) )
(seconds '() )
(seconds '(()) )
(seconds '((a)) )
(seconds '((a b) )
(seconds '((a b) (e 2)) )
(seconds '((a b c) (e 2 d)) )
```

insertR

```
(insertR 'a 'b '(a b e) )
(insertR 'a 'b '(a a a) )
(insertR 'a 'b '(a c e) )
(insertR 'a 'b '(c b e) )
(insertR 'a 'b '() )
(insertR 'a 'b '(aaa) )
```

Chapter 3 − 4

- Substr2 − two olds, replace with one new
- multirember − removes all occurances
- multiinsertR − insert to right of all occurances
- Chapter 4:
- add1, sub1
- o+, o−
- addtup − adds up the elements of a list
- tup+
- *, <, >, ^

Mystery Function

```
(define ???
  (lambda (n m)
    (cond
      ((< n m) 0)
      (else (add1 (??? (- n m) m)))
     )))
```