Scott Waldron
W00970369
CSCI301
Lab 6

Please provide the definition for four different, new functions from chapter 5 and 6. Also, give a brief description of the purpose of the function. Finally, include a sample implementation for each function.

#1 function (rember*)

```
(define rember*
  (lambda (a l)
    (cond
      ((null? l) (quote()))
      ((atom? (car l))
      (cond
        ((eq? (car l) a)
        (rember* a (cdr l)))
        (else (cons (car l)
          (rember* a (cdr l))))))
      (else (cons (rember* a (car l))
        (rember* a (cdr l)))))))
```

The purpose of rember* is to recur with the car as well as with the cdr. If the car is a sublist, it will remove the atom occurance from it. It is a deep rember rather than a surface.

```
a = apple
l = ((apple banana) apple (cherry apple))

(rember* a l) will return:
        ((banana) (cherry))
```

#2 function (insertR*)

```
(define insertR*
  (lambda (new old l)
    (cond
      ((null? l) quote()))
      ((atom? (car l))
      (cond
        ((eq? (car l) old)
        (cons old
          (cons new
            (insertR* new old cdr l))))))
      (else (cons (car l)
          (insertR* new old cdr l))))))
    (else (cons (insertR* new old (car l))
      (insertR* new old (cdr l))))))))
```

The purpose of insertR* is to recur with the car as well as with the cdr. If the car is a sublist, it will add new to the right of old. It is a deep insertR rather than a surface. It is similar to rember*

```
new = BOO
old = scream
l   = ((I scream) from the (top of the world) I scream)

(insertR* new old l) will return:
        ((I scream BOO) from the (top of the world) I scream BOO)
```

#3 function (occur*)

```
(define occur*
  (lambda (a l)
    (cond
      ((null? l) 0)
      ((atom? (car l))
      (cond
        ((eq? (car l) a)
          (add1 (occur* a (cdr l))))
      (else (occur* a (cdr l)))))
    (else(+ (occur* a (car l)) (occur* a (cdr l)))))))
```

The purpose of occur* is to recur with the car as well as the cdr. If the car is a sublist, it will add 1 for everytime a occurs in the sublist. It is another * so it asks 3 questions like the others.

```
a = the
l = (the (white dog) likes (the fat (cat and the fish)))

(occur* a l) will return:
        3
```

#4 function (member*)

```
(define member*
  (lambda (a l)
    (cond
      ((null? l) #f)
      ((atom? (car l))
      (or (eq? (car l) a)
        (member* a (cdr l))))
    (else (or (member* a (car l))
      (member* a (cdr l)))))))
```

The purpose of member* is to recur with the car as well as the cdr. If the car is a sublist, it search through and return true if a is a member of the list or any sublist.

```
a = truth
l = ((I only) speak (the(truth of (the matter))))

(member* a l) will return:
        #t
```