;;Homework 1.17

(define down

(lambda (lst)

(if (null? lst)

'()

(cons (list(car lst)) (down (cdr lst))))))

(define a '(1 2 3))

(define b '(a (b d) c d))

;;Homework 1.18

(define subSwapper

(lambda (s1 s2 sElement)

(if (symbol? sElement)

(cond

((eqv? sElement s1) s2)

((eqv? sElement s2) s1)

(else sElement))

(swapper s1 s2 sElement))))

(define swapper

(lambda (s1 s2 slist)

(if (null? slist)

'()

(cons

(subSwapper s1 s2 (car slist))

(swapper s1 s2 (cdr slist))))))

;;Homework 1.19

(define list-set

(lambda (lst n x)

(if (null? lst)

'()

(cond

( (= n 0) (cons x (cdr lst)))

(else (cons (car lst) (list-set (cdr lst) (- n 1) x)))))))

;;Homework 1.20

(define subCount

(lambda (x lst)

(if (symbol? lst)

(if (eqv? lst x) 1 0)

(count x lst))))

(define count

(lambda (x lst)

(if (null? lst)

0

(+ (subCount x (car lst)) (count x (cdr lst))))))