Allatoms: version 1

(define a1
  (lambda (lst)
    (cond ( (null? lst) '() )
      ( (= (length lst) 1) lst )
      ( else (cons (a1 (car lst))
                    (a1 (cdr lst))) )
    )
  )
)

1 ]=> (a1 '((b c)))
;Value 1: ((b c))

1 ]=> (a1 '((b c) d))
;The object b, passed as the first argument to length, is not the correct type.

Allatoms: version 2

(define a2
  (lambda (lst)
    (cond ( (null? lst) '() )
      ( (= (length lst) 1) lst )
      ( else (append (if (pair? (car lst))
                        (a2 (car lst))
                        (list (car lst)))
                    (a2 (cdr lst))) )
    )
  )
)

1 ]=> (a2 '((a (b c) d))
;Value 4: (a b c d)

1 ]=> (a2 '(((a) (c) ((d)) (e (f (g))) h)))
;Value 5: (a (c d (e (f (g)) h)))

1 ]=> (a2 '((b c)))
;Value 6: ((b c))

Allatoms: version 3

(define a3
  (lambda (lst)
    (cond ( (null? lst) '() )
      ( else (append (if (pair? (car lst))
                        (a3 (car lst))
                        (list (car lst)))
                    (a3 (cdr lst))) )
    )
  )
)

1 ]=> (a3 '((b c)))
;Value 7: (b c)

1 ]=> (a3 '((b c) d))
;Value 8: (a b c d)

1 ]=> (a3 '(((a) (c) ((d)) (e (f (g)) h))))
;Value 9: (a (c d e f g h))

Allatoms: version 4

(define a4
  (lambda (lst)
    (cond ( (null? lst) '() )
      ( (pair? lst) (append (a4 (car lst))
                             (a4 (cdr lst))) )
      ( else (list lst) )
    )
  )
)

This is simpler, but changes the specification of the procedure:

1 ]=> (a4 '(((a) (c) ((d)) (e (f (g)) h))))
;Value 10: (a c d e f g h)

1 ]=> (a4 '(a . b))
;Value 11: (a b)

1 ]=> (a4 'a)
;Value 12: (a)
Countatoms

(define countatoms
  (lambda (lst)
    (cond ((null? lst) 0)
          (else (+ (if (pair? (car lst))
                       (countatoms (car lst)) 1)
                  (countatoms (cdr lst))))))
  )

(define countall
  (lambda (lst)
    (cond ((null? lst) 0)
          ((pair? lst) (+ (countall (car lst))
                           (countall (cdr lst))))
          (else 1))
  )
)

Efficient version:

(define rev2
  (lambda (restsofar)
    (cond ((null? rest) sofar)
          (else (rev2 (cdr rest)
                       (cons (car rest) sofar))))))
)

A nicer interface.

(define my-reverse
  (lambda (lst)
    (rev2 lst')))
Efficient version:

(define most2
  (lambda (lst)
    (cond ((null? lst) 0)
          ((biggest (countatoms (car lst))
                       (most2 (cdr lst)))))
    )
  )
)

(define biggest
  (lambda (x y)
    (if (> x y) x y)
  )
)

In fact, Scheme has a built-in max procedure, so in this case we don’t need to write our own helper.

---

Let

When a helper procedure is not a natural solution to Pitfall 2, use let.

(let ((var1 value1)
       (var2 value2)
       ...
       (varg valuen)
   )
  ; Can now use var1 through varn

  ; var1 through varn now revert to having
  ; no value, or to their previous values if
  ; they had any.