```
;; Comprueba si un elemento x está en una lista "sin" sublistas

(define (pertenece? x lista)
  (cond
    ;; se comprueba si no es una lista
    ((not (list? lista))        #f)

    ;; se comprueba si es la lista vacía
    ((null? lista)              #f)

    ;; se comprueba si es igual al primer elemento
    ((equal? x (car lista))   #t)

    ;; se busca en el resto de la lista
    (else                       (pertenece? x (cdr lista)))
    )
  )
```

```
;; Comprueba si un elemento x está en una lista "con" sublistas

(define (pertenece-completo? x lista)
  (cond
    ;; se comprueba si no es una lista
    ((not (list? lista))    #f)
    ;; se comprueba si es la lista vacía
    ((null? lista)          #f)
    ;; se comprueba si el primer elemento es una sublista
    ((list? (car lista))
                (cond
                      ;; se comprueba si está en la sublista
                      ((pertenece-completo? x (car lista)) #t)
                      ;; se busca en el resto de la lista
                      (else    (pertenece-completo? x (cdr lista)))
                 )
     )

    ;; se comprueba si es igual al primer elemento
     ((equal? x (car lista)) #t)

    ;; se busca en el resto de la lista
    (else          (pertenece-completo? x (cdr lista)))
    )
  )
```

```
;; Esta función extrae todos los elementos atómicos o simples de una lista con sublistas
(define (elementos lista)
  (cond
    ;; se comprueba si no es una lista
    ((not (list? lista))    ())

    ;; se comprueba si es la lista vacía
    ((null? lista)          ())

    ;; se comprueba si el primer elemento es una sublista
    ((list? (car lista))
          ;; se concatenan las listas generadas por la primera sublista y el resto de la lista
                  (append
                      (elementos (car lista))
                      (elementos (cdr lista))
                  )
     )

    ;; se añade el primer elemento a la lista generada por el resto de la lista
    (else          (cons  (car lista)
                          (elementos (cdr lista))
                   )
    )
  )
)
```

```
;; Esta función cuenta todos los elementos atómicos de una lista "con" sublistas

(define (contar-elementos lista)
  (length (elementos lista))
  )
```

```
;; Esta función invierte todos los elementos atómicos de una lista "con" sublistas

(define (invertir lista)
  (cond
    ;; se comprueba si es una lista
    ((not (list? lista))    ())
    ;; se comprueba si es la lista vacía
    ((null? lista)          ())
    ;; se comprueba si el primer elemento es una sublista
    ((list? (car lista))
    ;; se concatenan las listas generadas por el resto de la lista y por la primera sublista
                        (append
                            (invertir (cdr lista))
                            (list (invertir (car lista)))
                        )
     )

    ;; se añade al final el primer elemento a la lista generada por el resto de la lista
    (else           (append
                            (invertir (cdr lista))
                            (list (car lista))
                    )
    )
  )
)
```