Write the tests first.
Each test should be as unique as possible.

;; merge: combines two sorted lists into one
;;   inputs: sorted lists L1 and L2
;;   outputs: a new list that contains all
;;            the elements of L1 and L2, in
;;            sorted order
;; NOTE: you must use recursion, not higher-order functions

(define (merge L1 L2)
  (sort (append L1 L2) <))
Converting units

1 ft = 12 in

1 in ≈ 1/40 m

1 min = 60 s

1 stokes = 1/10,000 m²/s

1 in ≈

40 in ≈

3 ⅓ ft ≈

1 ft/min ≈

20,000 ft stokes ≈

60 stokes/min ≈
Converting units

1 ft = 12 in
1 in ≈ 1/40 m
1 min = 60 s
1 stokes = 1/10,000 m²/s

1 in ≈ 1/40 m
40 in ≈ 1 m
3 ⅓ ft ≈ 1 m
1 ft/min ≈ 1/200 m/s
20,000 ft stokes ≈ ¾ m³/s
60 stokes/min ≈ 1/10,000 m²/s²
Normalization

There are two kinds ⇒ mutual recursion

(define unicalc-db
  (list
    (list 'foot   '(12.0 (inch)        ()))
    (list 'inch   '(0.025 (meter)       ()))
    (list 'minute '(60.0   (second)      ()))
    (list 'stokes '(0.0001 (meter meter) (second))))
)

> (assoc 'foot unicalc-db)
'(foot (12.0 (inch) ()))

> (assoc 'toe unicalc-db)
#f

(define (norm-unit unit)
  ...
)
(define (norm-QL QL) ...

Warning! Quantity lists here are not quite the same as in the assignment.