Here is a data definition for a ListOfNumbers, and two functions that process them:

;; A ListOfNumber is one of:
;; - empty
;; - (cons Number ListOfNumber)

;; add1-to-all : Number ListOfNumber -> ListOfNumber
;; Add the given number to all the elements of the given list
(define (add1-to-all l)
  (cond [(empty? l) l]
        [else (cons (add1 (first l))
                   (add1-to-all (rest l))))])

;; do-to-numbers : [Number -> Number] ListOfNumber -> ListOfNumber
;; Apply the given function to every element of the given list
(define (do-to-numbers f l)
  (cond [(empty? l) l]
        [else (cons (f (first l))
                   (do-to-numbers f (rest l)))]))

Provide a new implementation of add1-to-all which uses do-to-numbers.