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

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

;; remove-small : ListOfNumber -> ListOfNumber
;; remove all the numbers in the given list less than 10
(define (remove-small l)
  (cond 
    [(empty? l) l]
    [else (cond [(< (first l) 10) (remove-small (rest l))]
                 [else (cons (first l) (remove-small (rest l)))])])

;; remove-negative : ListOfNumber -> ListOfNumber
;; remove all the numbers in the given list less than 0
(define (remove-negative l)
  (cond 
    [(empty? l) l]
    [else (cond [(< (first l) 0) (remove-negative (rest l))]
                [else (cons (first l) (remove-negative (rest l)))])])
```

Circle the parts of the two functions that are different. Then design a new function which takes a *ListOfNumber* and a *Number* as an input, and removes all the numbers less than the given number from the list.