

# CSE 240 Introduction to Programming Languages

## Assignment 12 Merge Sort in Scheme

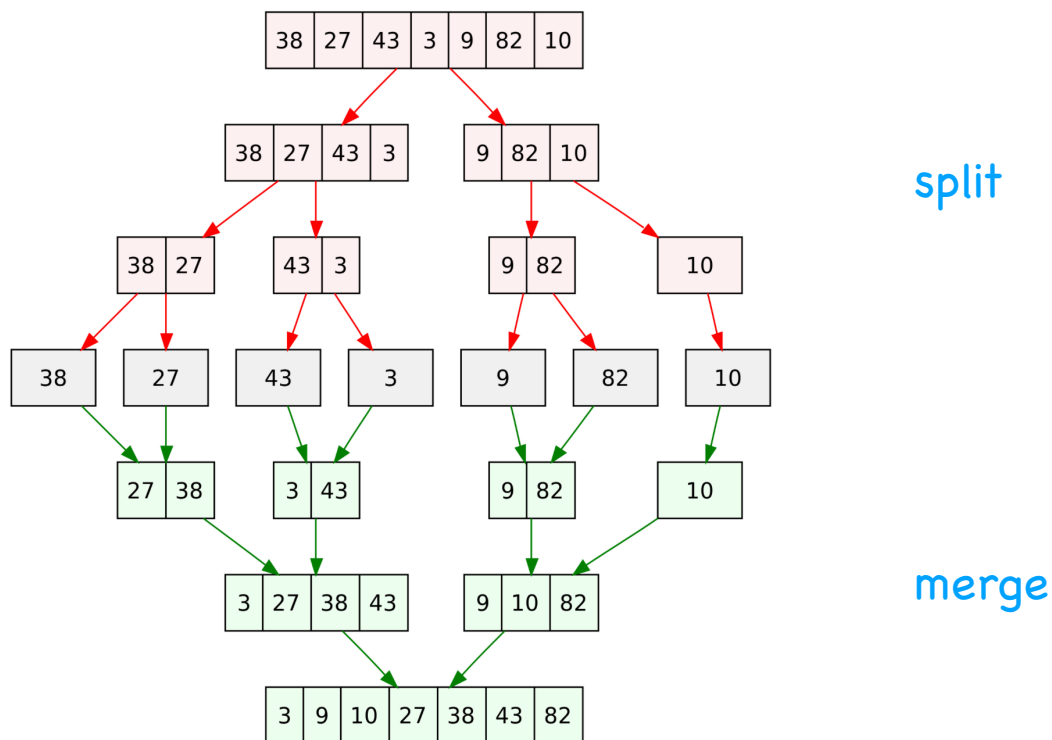
Due: Tuesday, April 9th at 11:59pm

Submit mergesort.rkt to Gradescope.



### Merge Sort

Merge sort is a general-purpose, comparison-based sorting algorithm that was invented by John von Neumann in 1945. It is a divide-and-conquer algorithm; the diagram below shows how it recursively splits the original list into roughly two halves and then merges these smaller lists back into a complete sorted list.



### Requirements

A Scheme program is needed to perform merge sort on a list of integers.


### Design and Implementation


A .rkt file is provided to guide your design and implementation. Code is provided for some of the procedures but you will need to complete the bodies of others. Your output should match that provided below.

```


(define L '(3 5 2 8 6 1 4 9 7))
(display "---- original list \n")
(display L) (newline)


; leftlist and rightlist
;-----
(define center (quotient (length L) 2))

(define (leftlist lst n)
  )

(define (rightlist lst n)
  )

(display "---- left and right lists\n")
(leftlist L center)
(rightlist L center)

; lefthalf and righthalf
;-----
(define (lefthalf lst)
  )

(define (righthalf lst)
  )

(display "---- left and right half\n")

(display "left halves\n")
(lefthalf L)
(lefthalf (lefthalf L))

(display "right halves\n")
(righthalf L)
(righthalf (righthalf L))

; merge
;-----
(define (merge lst1 lst2)
  (cond ((null? lst1) lst2)
        ((null? lst2) lst1)
        (else (if (< (car lst1) (car lst2))
                    
                    (cons (car lst2) (merge (cdr lst2) lst1))))))

```

```

(display "---- merge test 1\n")
(merge '(2 7 9) '(3 6 8))
(display "---- merge test 2\n")
(merge (leftlist L center) (rightlist L center))

```

```

; mergesort

```

```

;-----
(define (mergesort lst)
  (if (> (length lst) 1)
      (merge (mergesort (lefthalf lst))
              (mergesort (righthalf lst)))
      lst))

```

```

(display "---- mergesort test\n")
(mergesort L)

```

## Output

```

---- original list
(3 5 2 8 6 1 4 9 7)
---- left and right lists
(3 5 2 8)
(6 1 4 9 7)
---- left and right half
left halves
(3 5 2 8)
(3 5)
right halves
(6 1 4 9 7)
(4 9 7)
---- merge test 1
(2 3 6 7 8 9)
---- merge test 2
(3 5 2 6 1 4 8 9 7)
---- mergesort test
(1 2 3 4 5 6 7 8 9)
>

```