PROJECT DOCUMENTATION

COSC 352 – Organization of Programming Languages

Spring 2021

PROJECT 1

PROBLEM P

PROBLEM P

Write:

(a) In the Racket programming language, sort a linear list of numbers in increasing order – sort(l)

(b) At least three examples of the function sort(l)

Project Program sort.rkt

```
; Constructed Append function to add new list items

(define (append lhs rhs)
  (if (empty? lhs)
      rhs
      (cons (first lhs) (append (rest lhs) rhs))))


; Sort function sort(l)

(define (sort unsorted_arr [processed_arr (cdr unsorted_arr)]
[minimum_number (car unsorted_arr)] [sorted_arr (list)] [index 0]
[arr_length (length processed_arr)])
  (if (empty? processed_arr) (append sorted_arr (list minimum_number))
      (cond
        [(equal? index arr_length) (sort unsorted_arr (cdr processed_arr) (car
processed_arr) (append sorted_arr (list minimum_number)) 0 (- (length
processed_arr) 1))]
        [(> minimum_number (car processed_arr)) (sort unsorted_arr (append
(cdr processed_arr) (list minimum_number)) (car processed_arr) sorted_arr
(+ index 1) arr_length)]
        [else (sort unsorted_arr (append (cdr processed_arr) (list (car
processed_arr))) minimum_number sorted_arr (+ index 1) arr_length)]
        )
      )
  )
```

At Least 3 Test Examples

```
Welcome to DrRacket, version 7.9 [3m].
Language: racket, with debugging; memory limit: 128 MB.

> (sort '(1 -2 90 34 4))
'(-2 1 4 34 90)

> (sort '(-1 -2 -3 -4))
'(-4 -3 -2 -1)

> (sort '(7 1000 3 -100 5))
'(-100 3 5 7 1000)

> (sort '(1 5 1 5 198 19 3 3 198 10 4 10 49 20 1 3 30 2 48 2 8 2 1 3 2 10 18
37 28))
'(1 1 1 1 2 2 2 2 3 3 3 3 4 5 5 8 10 10 10 18 19 20 28 30 37 48 49 198 198)

> (sort '(1 49 19 2 288 19 19 293 19 1 192 91 92 8174 291 92 81 391
193718 18 81))
'(1 1 2 18 19 19 19 19 49 81 81 91 92 92 192 288 291 293 391 8174
193718)
```