Zoljargal Enkhbayar

Q2.

1.(defun partition (lst)

 (labels ((split-helper (lst left right)

     (cond

      ((null lst) (list left right)) ;; If empty, return both halves

      ((null (cdr lst)) (list (cons (car lst) left) right)) ;; If one element left

      (t (split-helper (cddr lst)

          (cons (car lst) left)

          (cons (cadr lst) right))))))

   (split-helper lst '() '())))

2. After partitioning, we recursively sort each half using mergesort function.

3. (defun merge (left right)

 (cond

  ((null left) right) ;; If left is empty, return right

  ((null right) left) ;; If right is empty, return left

  ((<= (car left) (car right)) ;; If first of left <= first of right, take from left

   (cons (car left) (merge (cdr left) right)))

  (t (cons (car right) (merge left (cdr right)))))) ;; Otherwise, take from right

4. (defun mergesort (lst)

 (if (or (null lst) (null (cdr lst))) ;; Base case: empty or one-element list

   lst

   (let* ((halves (partition lst))

      (left (mergesort (car halves)))

      (right (mergesort (cadr halves))))

    (merge left right))))

Q4.

a. In the beginning, the sorted list is empty (), and the unsorted list contains all the elements of the input list. At the end, the sorted list contains all the elements in ascending order, and the unsorted list will be empty. The process terminates when the unsorted list is empty. Sorted list will be the result.

b. The process is trivially accomplished when the unsorted list is empty. In this case, the sorted list is already the result.
In each pass of recursive call, we take the first element from the unsorted list and insert it into the correct position in the sorted list. The recursive call processes the remaining unsorted list.

c. In the insertion function, the insert function takes an element and inserts the element into the correct position in the sorted list. The last function in LISP can be used to find the last element of a list, but it is not necessary for insertion. We can just recursively traverse the list to find the correct position.
The insertion process terminates in two cases. If the new element is smaller than the first item in the sorted list, it is just put at the very front. Second, If the sorted list is empty, the new element will be the only one item in the list.