РК 2  
Функциональное программирование

Мирзоян С.А. ИУ7-65б

1 гр.

Дан смешанный, структурированный список (т.е. любой элемент списка может быть: символ, число или список).

Найти сумму четных чисел на всех уровнях заданного списка, лежащих в заданном интервале [a, b].

Добавить найденную сумму к исходному списку, в качестве К-ого элемента верхнего уровня или в конец. Если чисел нет, сообщить об этом.

Рекурсивный способ:

1. Реализация Рекурсиями
2. ; рекурсивная функция для подсчета суммы
3. (defun  func\_r (lst a b sum)
4. (cond
5. ((null lst) sum)
7. ((and   (numberp (car lst))
8. (evenp (car lst))
9. (or
10. (and   (>= (car lst) a) (<= (car lst)  b))
11. (and   (<= (car lst) a) (>= (car lst)  b))
12. )
13. )
15. (func\_r (cdr lst) a b (+ sum (car lst)))
17. )
19. ((listp  (car lst))  (func\_r (cdr lst) a b (func\_r (car lst) a b sum)))
21. (t (func\_r (cdr lst) a b sum))
22. )
23. )


27. ; обертка для главной рекурсивной функции
28. (defun  func (lst a b )
29. (func\_r lst a b 0)
30. )
32. ; рекурсивная вставка на k-ую  позицию
33. (defun insert\_r(lst res k new\_list)
34. (cond
35. ((and (null lst) (> k 0)) (reverse (cons res new\_list)))
36. ((null lst) (reverse new\_list))
37. ((eql  k 0) (insert\_r (cdr lst) res -1 (cons res new\_list)))
38. (T (insert\_r (cdr lst) res (- k 1) (cons (car lst) new\_list)))
39. )
40. )
41. ;Проверка наличия чия чисел
42. (defun check\_num\_r (lst help)
43. (cond
44. ((or (null lst) help)
45. help
46. )
47. ((numberp (car lst))
48. )
49. ((listp (car lst))
50. (check\_nums\_r (cdr lst) (check\_nums\_r (car lst) help))
51. )
52. (  (check\_nums\_r (cdr lst) help)    )
53. )
54. )
55. ;обертка  для проверки наличия  чисел
56. (defun check\_num (lst)
57. (check\_num\_r lst nil)
58. )
60. ; итоговая функция
61. (defun final\_func (lst a b k)
62. (cond
63. ((check\_num lst)
64. (insert\_r  lst (func lst a b)  k ())
65. )
66. (t (print “no numbers”))
67. )
68. )

71. (print (final\_func '(1 2 a b (3 4) (d 5) ((e) 6 8 9 10) )  2  4 2))

Функциональный способ:

1. (defun check\_cond (lst a b)
2. (mapcan #'(lambda (x)
3. (cond
4. (
5. (and
6. (numberp x)
7. (evenp x)
8. (or
9. (and (>= a x)(<= b x))
10. (and (<= a x)(>= b x))
11. )
12. )
13. (cons x nil)
14. )
15. (
16. (listp x)
17. (check\_cond  x a b)
18. )
19. )
20. ) lst
21. )
22. )
23. Функция cуммирующая список  чисел :
24. (defun get\_sum (lst a b)
25. (reduce #'+ (check\_cond  lst a b)))
27. Функция  вычисляет сумму  чисел и вставляет ее в конец списка
28. (defun final\_func  (lst a b)
29. (nconc lst (cons (get\_sum lst a b) nil))
30. )

Доп. Задание

1. (defun  (one\_level &optional acc)
2. (cond ((null w) acc)
3. ((atom w) (cons w acc))
4. ((one\_level (car w) (one\_level (cdr w) acc)))))
6. (defun del-num (lst)
7. (cond ((null lst) nil)
8. ((numberp (car lst)) (del-num (cdr lst)))
9. (t (cons (car lst) (del-num (cdr lst))))))