
предприятие (организация)

УДК _____

№ госрегистрации _____

Инв. № _____

УТВЕРЖДАЮ

головной исполнитель НИР

« _____ » _____ 2020 г.

ОТЧЁТ
О НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ

наименование и шифр НИР

(промежуточный)
шифр — _____

ВВЕДЕНИЕ

Целью работы является приобретение навыков организации рекурсии.

Задачи работы: изучить способы организации хвостовой, дополняемой, множественной, взаимной рекурсии и рекурсии более высокого порядка в Lisp.

1 Теоретические сведения

— Способы организации повторных вычислений в Lisp: функционалы, рекурсия.

— Что такое рекурсия? Классификация рекурсивных функций в Lisp: рекурсия — это ссылка на определяемый объект во время его определения; классификация: простая рекурсия - один рекурсивный вызов в теле; рекурсия первого порядка - рекурсивный вызов встречается несколько раз; взаимная рекурсия - используется несколько функций, рекурсивно вызывающих друг друга.

— Различные способы организации рекурсивных функций и порядок их реализации: хвостовая (результат формируется не на выходе из рекурсии, а на входе в рекурсию, все действия выполняя до ухода на следующий шаг рекурсии), по нескольким параметрам, дополняемая (при обращении к рекурсивной функции используется дополнительная функция не в аргументе вызова, а вне его), множественная (на одной ветке происходит сразу несколько рекурсивных вызовов).

— Способы повышения эффективности реализации рекурсии: применять хвостовую организацию рекурсии.

2 Практическая часть

2.1 Задание №1

Условие:

Пусть list-of-lists список, состоящий из списков. Написать функцию, которая вычисляет сумму длин всех элементов list-of-lists.

В литинге 2.1 приведен текст решения данной задачи на языке Common Lisp.

Листинг 2.1 — Задание №1

```
1 (defun lvl2len (lol)
2   (cond ((null lol) 0)
3         (t (+ (length (car lol))
4               (lvl2len (cdr lol))))))
5
6 (lvl2len nil) ; 0
7 (lvl2len '((1 2) (3 4))) ; 4
```

2.2 Задание №2

Условие:

Написать рекурсивную версию функции вычисления суммы чисел заданного списка.

В литинге 2.2 приведен текст решения данной задачи на языке Common Lisp.

Листинг 2.2 — Задание №2

```
1 (defun reg-add (lst)
2   (cond ((null lst) 0)
3         (t (+ (cond ((numberp (car lst)) (car lst))
4                     (t 0))
5               (reg-add (cdr lst))))))
6
7 (reg-add '(2 4 6)) ; 12
8 (reg-add '(2 4 word 6 nil)) ; 12
```

```
9 (reg-add '(word nil)) ; 0
10 (reg-add nil) ; 0
```

2.3 Задание №3

Условие:

Написать рекурсивную версию функции `nth`.

В литинге 2.3 приведен текст решения данной задачи на языке Common Lisp.

Листинг 2.3 — Задание №3

```
1 (defun recnth (idx lst)
2   (cond ((or (= 0 idx)
3              (null lst)) (car lst))
4         (t (recnth (- idx 1) (cdr lst)))))
5
6 (recnth 0 '(1 2 3)) ; 1
7 (recnth 2 '(1 2 3)) ; 3
8 (recnth 1 '(1 2 3)) ; 2
9 (recnth 5 '(1 2 3)) ; NIL
```

2.4 Задание №4

Условие:

Написать рекурсивную функцию, которая возвращает `t` когда все элементы списка нечетные.

В литинге 2.4 приведен текст решения данной задачи на языке Common Lisp.

Листинг 2.4 — Задание №4

```
1 (defun alloddr (lst)
2   (cond ((null lst) t)
3         ((evenp (car lst)) nil)
4         (t (alloddr (cdr lst)))))
5
```

```
6 (alloddr '(1 2 3)) ; nil
7 (alloddr '(1 3 5)) ; nil
```

2.5 Задание №5

Условие:

Написать рекурсивную функцию, относящуюся к хвостовой рекурсии с одним тестом завершения, которая возвращает последний элемент списка-аргумента.

В литинге 2.5 приведен текст решения данной задачи на языке Common Lisp.

Листинг 2.5 — Задание №5

```
1 (defun latest (lst)
2   (cond ((null (cdr lst)) (car lst))
3         (t (latest (cdr lst)))))
4
5 (latest nil) ; NIL
6 (latest '(1)) ; 1
7 (latest '(1 2 3 (4 5))) ; (4 5)
```

2.6 Задание №6

Условие:

Написать рекурсивную функцию, относящуюся к дополняемой рекурсии с одним тестом завершения, которая вычисляет сумму всех чисел:

- а) от 0 до n-ого аргумента функции;
- б) от n-аргумента функции до последнего ≥ 0 ;
- в) от n-аргумента функции до r-аргумента с шагом d.

В литинге 2.6 приведен текст решения данной задачи на языке Common Lisp.

Листинг 2.6 — Задание №6

```

1  ;;; 1)
2  (defun sum-first (nums n)
3    (cond ((or (= 0 n)
4              (null nums)) 0)
5          (t (+ (car nums)
6                (sum-first (cdr nums) (- n 1)))))
7
8  ;;; 2)
9  (defun help1 (nums n)
10   (cond ((or (null nums) (= 0 n)) 0)
11         (t (+ (car nums)
12               (help1 (cdr nums) (- n 1)))))
13
14 (defun help2 (nums n)
15   (cond ((or (null nums) (= 0 n)) 0)
16         ((> (car nums) 0) (help1 nums n))
17         (t (help2 (cdr nums) (- n 1)))))
18
19 (defun sum-to-last-pos (nums n)
20   (help2 (reverse nums) (- (length nums) n)))
21
22 ;;; 3)
23 (defun sum-range (nums lo hi h)
24   (cond ((or (= 0 hi)
25             (null nums)) 0)
26         (t (+ (sum-range (cdr nums)
27                         (cond ((= lo 0) h)
28                               (t (- lo 1)))
29                         (- hi 1)
30                         h)
31             (cond ((> lo 0) 0)
32                   (t (car nums))))))
33
34 (setq nums '(0 1 2 3 4 5 6 7 8 9))
35
36 (sum-first nums 5) ; 10
37 (sum-first nums 0) ; 0
38 (sum-first nums 1) ; 0
39 (sum-first nums 2) ; 1
40 (sum-first nums 10) ; 45
41
42 (sum-to-last-pos nums 5) ; 35
43 (sum-to-last-pos '(-1 2 3 4 -1) 2) ; 7
44 (sum-to-last-pos (concatenate 'list nums '(-10 5 -7 -9 -2)) 5) ; 30
45
46 (sum-range nums 1 9 2) ; 12

```

```
47 (sum-range nums 0 10 0) ; 45
48 (sum-range nums 0 10 1) ; 20
49 (sum-range nums 1 10 1) ; 25
```

2.7 Задание №7

Условие:

Написать рекурсивную функцию, которая возвращает последнее нечетное число из числового списка.

В литинге 2.7 приведен текст решения данной задачи на языке Common Lisp.

Листинг 2.7 — Задание №7

```
1 (defun help1 (lst lodd)
2   (cond ((null lst) lodd)
3         (t (help1 (cdr lst)
4                   (cond ((oddp (car lst)) (car lst))
5                         (t lodd))))))
6
7 (defun latestst-odd (lst)
8   (help1 lst nil))
9
10 (latestst-odd nil) ; NIL
11 (latestst-odd '(2 4 6 8)) ; NIL
12 (latestst-odd '(1 2 3 4 5 6 7 8 9 -1 10)) ; -1
```

2.8 Задание №8

Условие:

Используя cons-дополняемую рекурсию с одним тестом завершения, написать функцию которая получает как аргумент список чисел, а возвращает список квадратов этих чисел в том же порядке.

В литинге 2.8 приведен текст решения данной задачи на языке Common Lisp.

Листинг 2.8 — Задание №8

```
1 (defun sqr-nums (nums)
2   (cond ((null nums) nil)
3         (t (cons (* (car nums) (car nums))
4                   (sqr-nums (cdr nums))))))
5
6 (sqr-nums nil) ; NIL
7 (sqr-nums '(1 2 -3 4 5)) ; (1 4 9 16 25)
```

2.9 Задание №9

Условие:

Написать функцию, которая:

- а) выбирает из списка чисел все чётные;
- б) выбирает из списка чисел все нечётные;
- в) суммирует все четные числа списка;
- г) суммирует все нечетные числа списка.

В листинге 2.9 приведен текст решения данной задачи на языке Common Lisp.

Листинг 2.9 — Задание №9

```
1 (defun filter (predicate lst)
2   (cond ((null lst) nil)
3         (t (cond ((funcall predicate (car lst))
4                   (cons (car lst) (filter predicate (cdr lst))))
5                 (t (filter predicate (cdr lst))))))
6
7 (defun select-odd (nums)
8   (filter #'oddp nums))
9
10 (defun select-even (nums)
11   (filter #'evenp nums))
12
13 (defun sum-all-odd (nums)
14   (reduce #'+ (select-odd nums)))
15
16 (defun sum-all-even (nums)
```

```
17    (reduce #'+ (select-even nums)))
18
19    (setq digits '(0 1 2 3 4 5 6 7 8 9))
20
21    (select-odd digits) ; (1 3 5 7 9)
22    (sum-all-odd digits) ; 25
23
24    (select-even digits) ; (0 2 4 6 8)
25    (sum-all-even digits) ; 20
```