

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Сибирский государственный университет телекоммуникаций и информатики»
(СибГУТИ)

Кафедра прикладной математики и кибернетики

Лабораторная работа №2
по дисциплине «Функциональное и логическое программирование»
Бригада №4

Выполнили :
студенты группы ИП-811
Адов А.С.
Бобова И.А.
Работу проверил:
Галкина М.Ю.

Новосибирск
2020 г.

Задание:

4) Добавляющую заданное параметром x число к каждому числовому элементу списка. Например, $x=3$, $L=(a -1 6 v 3) \rightarrow (a 2 9 v 6)$.

14) Преобразующую список в множество (для повторяющихся элементов должно оставаться последнее вхождение в список). Например, $(a b a a c c) \rightarrow (b a c)$.

24) Переставляющую элементы списка таким образом, чтобы одинаковые элементы оказались рядом. Сортировку не использовать! Например, $(1 5 2 1 4 3 1 2 4 5 4) \rightarrow (1 1 1 5 5 2 2 4 4 4 3)$.

Листинг программы:

```
; Adov_&_Bobova; laboratory_work_2; variant_4
```

```
;task_4
```

```
(defun add (s a)
  (mapcar (lambda (x) (if (numberp x) (+ x a) x)) s))
(add '(a -1 6 v 3) 3)
```

```
;task_14
```

```
(defun convert(s)
  (cond ((null s) nil)
        ((member (car s) (cdr s)) (convert(cdr s)))
        (t (cons (car s) (convert (cdr s))))))
(convert '(a b a a c c))
```

```
;task_24
```

```
(defun task (s)
  (cond
    ((null s) nil)
    (t (append (remove-if-not (lambda (x) (equal (car s) x)) s) (task (remove (car s) s))))))
(task '(1 5 2 1 4 3 1 2 4 5 4))
```

Результат работы программы:

```
LispIDE - C:\Users\ASUS\Desktop\functional programming\lab2.lisp
File Edit Search View Settings Window Help

lab1.lisp lab2.lisp
1 ; Adou_B_Bobova; laboratory_work_2; variant_4
2
3 ;task_4
4 (defun add (s a)
5   (mapcar (lambda (x) (if (numberp x) (+ x a) x)) s))
6 (add '(a -1 6 0 3) 3)
7
8 ;task_14
9 (defun convert(s)
10  (cond ((null s) nil)
11        ((member (car s) (cdr s)) (convert(cdr s)))
12        (t (cons (car s) (convert (cdr s))))))
13 (convert '(a b a a c c))
14
15 ;task_24
16 (defun task (s)
17   (cond
18     ((null s) nil)
19     (t (append (remove-if-not (lambda (x) (equal (car s) x)) s) (task (remove (car s) s))))))
20 (task '(1 5 2 1 4 3 1 2 4 5 4))

[1]>
ADD
[2]>
(A 2 9 0 6)
[3]>
CONVERT
[4]>
(B A C)
[5]>
TASK
[6]>
(1 1 1 5 5 2 2 4 4 4 4 3)
[7]>

Ready
```