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 №2

Выполнили:

студент группы ИВ-823

Лозовой Владислав Александрович

ФИО студента

студент группы ИВ-823

Шиндель Эдуард Дмитриевич

ФИО студента

Работу проверил:

ассистент кафедры Агалаков А.А.

ФИО преподавателя

Новосибирск 2020 г.

# ОГЛАВЛЕНИЕ

[ОГЛАВЛЕНИЕ 2](#_Toc52186895)

[ЗАДАНИЕ 3](#_Toc52186896)

[ДЕМОНСТРАЦИЯ РАБОТЫ ПРОГРАММЫ 3](#_Toc52186897)

[ВЫВОД 3](#_Toc52186898)

[ПРИЛОЖЕНИЕ 5](#_Toc52186899)

# 

# 

# ЗАДАНИЕ

Определить функцию:

1. Преобразующую список в "двойной" список таким образом, чтобы каждый элемент удвоился.

Например, (1 a b 3) –> (1 1 a a b b 3 3)

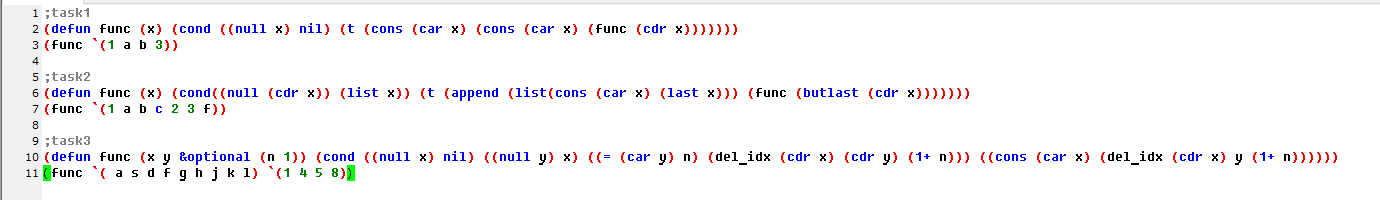
1. Возвращающую список из пар соседних элементов, одинаково отстоящих от начала и конца списка.

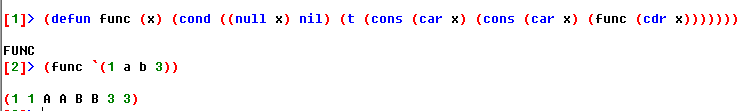
Например, (1 a b с 2 3 f) –> ((1 f) (a 3) (b 2) (с))

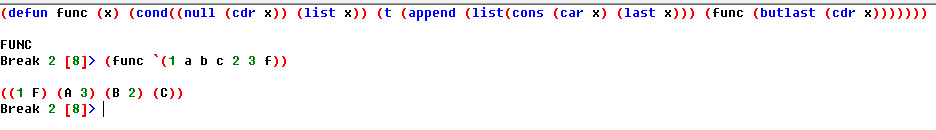
1. Удаляющую элементы из первого списка с номерами из второго списка (второй список упорядочен по возрастанию, нумерация элементов должна начинаться с 1).

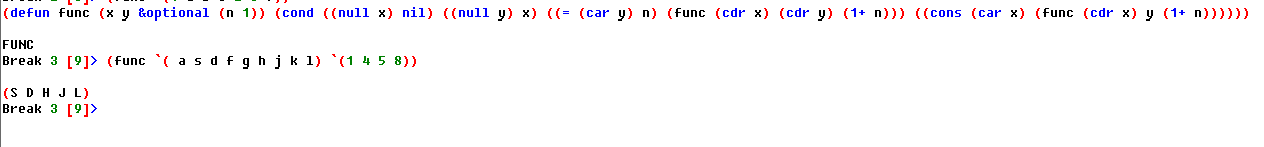
Например, L1=(a s d f g h j k l), L2=(1 4 5 8) –> (s d h j l)

# ДЕМОНСТРАЦИЯ РАБОТЫ ПРОГРАММЫ









**ВЫВОД**

Было выполнено 3 задачи в LISP по написанию функций без использования set, let и setq. \

Написаны функции:

1. Для преобразования списка в двойной список
2. Возвращающая список из пар соседних элементов, отстоящих от начала и конца списка
3. Удаляющая элементы из первого списка с номерами из второго

# ПРИЛОЖЕНИЕ

;task1

(defun func (x) (cond ((null x) nil) (t (cons (car x) (cons (car x) (func (cdr x)))))))

(func `(1 a b 3))

;task2

(defun func (x) (cond((null (cdr x)) (list x)) (t (append (list(cons (car x) (last x))) (func (butlast (cdr x)))))))

(func `(1 a b с 2 3 f))

;task3

(defun func (x y &optional (n 1)) (cond ((null x) nil) ((null y) x) ((= (car y) n) (func (cdr x) (cdr y) (1+ n))) ((cons (car x) (func (cdr x) y (1+ n))))))

(func `( a s d f g h j k l) `(1 4 5 8))