**Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский государственный технический университет имени Н.Э. Баумана**

**(национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

# Отчет по лабораторной работе №7 по курсу

**«Функциональное и логическое программирование»**

**Тема** Рекурсивные функции

**Студент** Климов И.С.

**Группа** ИУ7-62Б

**Оценка (баллы)**

**Преподаватели** Толпинская Н.Б., Строганов Ю.В.

Москва — 2022 г.

## **Задание 1 (1)**

Написать хвостовую рекурсивную функцию my-reverse, которая развернет верхний уровень своего списка-аргумента lst.

**Решение**

(defun move\_to (lst result)

(cond ((null lst) result)

(T (move\_to (cdr lst) (cons (car lst) result)))))

(defun my\_reverse (lst)

(move\_to lst ()))

## **Задание 2 (3)**

Написать функцию, которая возвращает первый элемент списка -аргумента, который сам является непустым списком.

**Решение**

(defun find\_list (lst element)

(cond ((or (null lst) element) element)

((and (listp (car lst)) (> (length lst) 0))

(find\_list (cdr lst) (car lst)))

(T (find\_list (cdr lst) element))))

(defun find\_first\_list (lst)

(find\_list lst NIL))

## **Задание 3 (4)**

Написать функцию, которая выбирает из заданного списка только те числа, которые больше 1 и меньше 10. (Вариант: между двумя заданными границами).

**Решение**

(defun between (lst result left right)

(cond ((null lst) result)

((and (> (car lst) left) (< (car lst) right))

(between (cdr lst) (append result (list (car lst))) left right))

(T (between (cdr lst) result left right))))

(defun between\_1\_and\_10 (lst)

(between lst () 1 10))

## **Задание 4 (7)**

Напишите рекурсивную функцию, которая умножает на заданное число-аргумент все числа из заданного списка-аргумента, когда

1. все элементы списка – числа,
2. элементы списка – любые объекты.

**Решение**

(defun mul\_by\_number\_v1 (lst num &optional (result NIL))

(cond ((null lst) result)

(T (mul\_by\_number\_v1 (cdr lst) num (cons (\* (car lst) num) result)))))

(defun mul\_by\_number\_v2 (lst num &optional (result NIL))

(cond ((null lst) result)

((numberp (car lst))

(mul\_by\_number\_v2 (cdr lst) num (cons (\* (car lst) num) result)))

(T (mul\_by\_number\_v2 (cdr lst) num (cons (car lst) result)))))

## **Задание 5 (8)**

Напишите функцию, select-between, которая из списка-аргумента, содержащего только числа, выбирает только те, которые расположены между двумя указанными границами-аргументами и возвращает их в виде списка (упорядоченного по возрастанию списка чисел (+ 2 балла)).

**Решение**

(defun my\_remove (x lst &optional (deleted 0) (result NIL))

(cond ((null lst) result)

((and (= deleted 0) (equal x (car lst)))

(my\_remove x (cdr lst) 1 result))

(T (my\_remove x (cdr lst) deleted (append result (list (car lst)))))))

(defun my\_max (lst &optional (max NIL))

(cond ((null lst) max)

((or (null max) (> (car lst) max)) (my\_max (cdr lst) (car lst)))

(T (my\_max (cdr lst) max))))

(defun my\_sort (lst &optional result)

(cond ((null lst) result)

(T (my\_sort (my\_remove (my\_max lst) lst)

(cons (my\_max lst) result)))))

(defun select\_between (lst left right)

(cond ((> left right) (select\_between lst right left))

(T (select\_between lst left right))))

## **Задание 6 (8)**

Написать рекурсивную версию (с именем rec-add) вычисления суммы чисел заданного списка:

1. одноуровнего смешанного,
2. структурированного.

**Решение**

(defun rec\_add\_v1 (lst &optional (sum 0))

(cond ((null lst) sum)

((numberp (car lst)) (rec\_add\_v1 (cdr lst) (+ sum (car lst))))

(T (rec\_add\_v1 (cdr lst) sum))))

(defun rec\_add\_v2 (lst)

(cond ((numberp lst) lst)

((atom lst) 0)

(T (+ (rec\_add\_v2 (car lst)) (rec\_add\_v2 (cdr lst))))))

## **Задание 7 (9)**

Написать рекурсивную версию с именем recnth функции nth.

**Решение**

(defun recnth (n lst)

(cond ((= n 0) (car lst))

(T (recnth (- n 1) (cdr lst))))

## **Задание 8 (10)**

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

**Решение**

(defun my\_oddp (x)

(and (numberp x) (= (mod x 2) 1)))

(defun all\_odd (lst)

(cond ((null lst) T)

((my\_oddp (car lst)) (all\_odd (cdr lst)))

(T NIL)))

## **Задание 9 (11)**

Написать рекурсивную функцию, которая возвращает первое нечетное число из списка (структурированного), возможно создавая некоторые вспомогательные функции.

**Решение**

(defun first\_odd (lst)

(cond ((my\_oddp lst) lst)

((atom lst) NIL)

(T (or (first\_odd (car lst)) (first\_odd (cdr lst))))))

## **Задание 10 (12)**

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.

**Решение**

(defun square\_list (lst)

(cond ((null lst) NIL)

(T (cons (\* (car lst) (car lst)) (square\_list (cdr lst)))))