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»

Отчёт
по лабораторной работе №7

Название: Рекурсивные функции

Дисциплина: Функциональное и логическое программирование

Студент ИУ7-64Б _____ Л.Е.Тартыков
(Группа) (Подпись, дата) (И.О. Фамилия)

Преподаватель _____ Н.Б.Толпинская
(Подпись, дата) (И.О. Фамилия)

Преподаватель _____ Ю.В.Строганов
(Подпись, дата) (И.О. Фамилия)

Москва, 2022

1 Практические задания

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

Листинг 1.1 – Задание 1

```
(defun my_reverse (lst)
    (move_to_front lst ())
)

(defun move_to_front (lst result)
    (cond ((null lst) result)
          (T (move_to_front (cdr lst) (cons (car lst) result)) )
    )
)
```

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

Листинг 1.2 – Задание 2

```
(defun return_first_list (lst)
    (cond ((null lst) nil)
          ((not (and (listp (car lst)) (> (length (car lst)) 0) )) (
             return_first_list (cdr lst)) )
          (T (car lst))
    )
)
```

1.3 Написать функцию, которая выбирает из заданного списка только те числа, которые больше 1 и меньше 10.

Листинг 1.3 – Задание 3

```
(defun return_in_range (lst)
    (cond ((null lst) nil)
          ((and (> (car lst) 1) (< (car lst) 10)) (cons (car lst) (
             return_in_range (cdr lst))))
          (T (return_in_range (cdr lst)))
    )
)
```

1.4 Напишите рекурсивную функцию, которая умножает на заданное число-аргумент все числа из заданного списка-аргумента, когда а) все элементы списка – числа, 6) элементы списка – любые объекты.

Листинг 1.4 – Задание 4

```
(defun multiply_by_number_1 (lst number)
    (cond ((null lst) nil)
          ((cons (* number (car lst)) (multiply_by_number_1 (cdr lst) number) ))
    )
)

(defun multiply_by_number_2 (lst number)
    (cond ((null lst) nil)
          ((numberp (car lst)) (cons (* number (car lst)) (multiply_by_number_2
             (cdr lst) number)))
          (T (cons (car lst) (multiply_by_number_2 (cdr lst) number)))
    )
)
```

1.5 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.

Листинг 1.5 – Задание 5

```
(defun select_between (lst board_left board_right)
    (bubble_sort_asc (select lst board_left board_right))
)

(defun select (lst board_left board_right)
    (cond ((null lst) nil)
          ((and (>= (car lst) board_left) (<= (car lst) board_right))
          (cons (car lst) (select (cdr lst) board_left board_right))
          )
          (T (select (cdr lst) board_left board_right))
    )
)

(defun bubble_sort_asc (lst)
    (cond ((atom (cdr lst)) lst)
           (T (bubble (cons (car lst) (bubble_sort_asc (cdr lst)))))
    )
)

(defun bubble (lst)
    (cond ((atom (cdr lst)) lst)
          ((> (car lst) (cadr lst))
          (cons (cadr lst) (bubble (cons (car lst) (cddr lst)) ))
          )
          (T lst)
    )
)
```

1.6 Написать рекурсивную версию (с именем rec-add) вычисления суммы чисел заданного списка: а) одноуровнего смешанного, б) структурированного.

Листинг 1.6 – Задание 6

```
(defun rec_add_1 (lst)
    (let ((sum 0))
         (calc_sum lst sum)
    )
)

(defun calc_sum (lst sum)
    (cond ((null lst) sum)
          ((numberp (car lst)) (calc_sum (cdr lst) (+ sum (car lst))))
          (T (calc_sum (cdr lst) sum))
    )
)

(defun rec_add_2 (lst)
    (calc_sum_structure_lst lst 0)
)

(defun calc_sum_structure_lst (lst sum)
    (cond ((null lst) sum)
          ((numberp (car lst)) (calc_sum_structure_lst (cdr lst) (+ sum (car lst
              ))))
          ((listp (car lst)) (+ (calc_sum_structure_lst (car lst) sum) (
              calc_sum_structure_lst (cdr lst) 0)))
          (T (calc_sum_structure_lst (cdr lst) sum))
    )
)
```

1.7 Написать рекурсивную версию с именем recnth функции nth.

Листинг 1.7 – Задание 7

```
(defun rec_nth (lst position)
    (cond ((not (check_position lst position)) nil)
          (T (find_elem lst position 0))
    )
)

(defun check_position (lst position)
    (cond ((> position (- (length lst) 1)) nil)
          (T T)
    )
)

(defun find_elem (lst position index)
    (cond ((= index position) (car lst))
          (T (find_elem (cdr lst) position (+ index 1)))
    )
)
```

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

Листинг 1.8 – Задание 8

```
(defun all_odd (lst)
    (cond ((null lst) T)
          ((oddp (car lst)) (all_odd (cdr lst)))
          (T nil)
    )
)
```

1.9 Написать рекурсивную функцию, которая возвращает первое нечетное число из списка (структурированного).

Листинг 1.9 – Задание 9

```
(defun find_first_odd (lst)
    (cond ((null lst) nil)
          ((numberp (car lst)) (cond ((oddp (car lst)) (car lst))
                                     (T (find_first_odd (cdr lst)))))
          )
          ((listp (car lst)) (or (find_first_odd (car lst)) (find_first_odd (cdr
              lst))))
          (T (find_first_odd (cdr lst)))
    )
)
```

1.10 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.

Листинг 1.10 – Задание 10

```
(defun square_lst_elems (lst)
    (cond ((null lst) nil)
          (T (cons (* (car lst) (car lst)) (square_lst_elems (cdr lst))))
    )
)
```