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 по курсу

«Функциональное и логическое программирование»

Тема Рекурсивные функции.

Студент Сироткина П.Ю.

Группа ИУ7-66Б

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

Москва — 2022 г.

Практические задания

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

```
(defun my_reverse(lst)
    (move_to lst Nil)
)

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

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

```
(defun first_sublist(lst)
    (cond
        ((null lst) Nil)
        ((atom (car lst)) (first_sublist (cdr lst)))
        (T (caar lst))
    )
)
```

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

```
(defun select_between(lst)
    (cond
        ((null lst) Nil)
        ((< 1 (car lst) 10) (cons (car lst) (select_between (cdr lst))))
        (T (select_between (cdr lst)))
    )
)
```

4. 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.

```
(defun mul_num(lst mul)
  (cond
      ((null lst) Nil)
      (T (cons (* (car lst) mul) (mul_num (cdr lst) mul)))
  )
)

(defun mul_all(lst mul)
  (cond
      ((null lst) Nil)
      ((numberp (car lst)) (cons (* (car lst) mul) (mul_all(cdr lst) mul)))
      ((atom (car lst)) (cons (car lst) (mul_all (cdr lst) mul)))
      (T (cons (mul_all (car lst) mul) (mul_all (cdr lst) mul)))
  )
)
```

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.

```
(defun select_between(lst left right)
    (cond
        ((null lst) Nil)
        ((< left (car lst) right)
                  (cons (car lst) (select_between (cdr lst) left right)))
        (T (select_between (cdr lst) left right))
    )
)
```

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

1. Одноуровнего смешанного;
2. Структурированного.

```
;1
(defun rec_add(lst) (helper lst 0))

(defun helper(lst acc)
    (cond
        ((null lst) acc)
        ((numberp (car lst)) (helper (cdr lst) (+ acc (car lst))))
        (T (helper (cdr lst) acc))
    )
)

;2
(defun rec_add(lst) (helper lst 0))

(defun helper(lst acc)
    (cond
        ((null lst) acc)
        ((numberp (car lst)) (helper (cdr lst) (+ acc (car lst))))
        ((listp (car lst)) (+ (helper (car lst) 0) (helper (cdr lst) acc)))
        (T (helper (cdr lst) acc))
    )
)
```

7. Написать рекурсивную версию с именем recnth функции nth.

```
(defun recnth(n lst)
    (cond
        ((null lst) Nil)
        ((= 0 n) (car lst))
        (T (recnth (- n 1) (cdr lst)))))
```

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

```
(defun allodd(lst)
    (cond
```

```
        ((null lst) T)
        ((oddp (car lst)) (allodd (cdr lst)))
        (T Nil)
    )
)
```

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

```
(defun first_atom(lst)
    (cond
        ((atom lst) lst)
        (T (first_atom (car lst)))
    )
)

(defun first_odd(lst)
    (cond
        ((null lst) Nil)
        ((oddp (first_atom lst)) (first_atom lst))
        (T (first_odd (cdr lst)))
    )
)
```

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.

```
(defun squares(lst)
    (cond
        ((null lst) Nil)
        (T (cons ((lambda (x) (* x x)) (car lst)) (squares (cdr lst))))
    )
)
```