



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»

Студент _____ Маслова Марина Дмитриевна
Группа _____ ИУ7-63Б
Оценка (баллы) _____
Преподаватель _____ Толпинская Наталья Борисовна
Преподаватель _____ Строганов Юрий Владимирович

2022 г.

1 Практические задания

1.1 Задание №1

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

```
1 (defun move-to (lst res)
2   (cond ((null lst) res)
3         (T (move-to (cdr lst) (cons (car lst) res)))))
4
5 (defun my-reverse (lst)
6   (move-to lst ()))
```

1.2 Задание №2

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

```
1 (defun get-first-list-el (lst)
2   (cond ((null lst) Nil)
3         ((and (car lst) (listp (car lst))) (caar lst))
4         (T (get-list (cdr lst)))))
```

1.3 Задание №3

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

```
1 (defun sba (lst begin end res)
2   (cond ((null lst) res)
3         ((and (numberp (car lst))
4               (or (< begin (car lst) end) (< end (car lst) begin)))
5         (sba (cdr lst) begin end (cons (car lst) res)))
6         ((atom (car lst)) (sba (cdr lst) begin end res))
7         (T (sba (cdr lst) begin end (sba (car lst) begin end res)))))
8
9 (defun select-between-all (lst begin end)
10  (sba lst begin end ()))
```

1.4 Задание №4

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

1. все элементы списка — числа,

```
1 (defun mul-nums-res (lst num res)
2   (cond ((null lst) res)
3         (T (mul-nums-res (cdr lst) num (cons (* (car lst) num) res)))))
4
5 (defun mul-nums (lst num)
6   (mul-nums-res lst num ()))
```

2. элементы списка — любые объекты.

```
1 (defun mn (lst num res)
2   (cond ((null lst) res)
3         ((numberp (car lst))
4          (mn (cdr lst) num (cons (* (car lst) num) res)))
5         ((atom (car lst)) (mn (cdr lst) num (cons (car lst) res)))
6         (T (mn (cdr lst) num (cons (mn (car lst) num ()) res)))))
7
8 (defun mul-num (lst num)
9   (mn lst num ()))
```

1.5 Задание №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.

```
1 (defun sb (lst begin end res)
2   (cond ((null lst) res)
3         ((or (< begin (car lst) end) (< end (car lst) begin))
4          (sb (cdr lst) begin end (cons (car lst) res)))
5         (T (sb (cdr lst) begin end res))))
6
7 (defun select-between (lst begin end)
8   (sb lst begin end ()))
```

1.6 Задание №6

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

1. одноуровневого смешанного;

```
1 (defun ra (lst res)
2   (cond ((null lst) res)
3         ((numberp (car lst)) (ra (cdr lst) (+ (car lst) res)))
4         (T (ra (cdr lst) res))))
5
6 (defun rec-add (lst)
7   (ra lst 0))
```

2. структурированного.

```
1 (defun ra (lst res)
2   (cond ((null lst) res)
3         ((numberp (car lst)) (ra (cdr lst) (+ (car lst) res)))
4         ((atom (car lst)) (ra (cdr lst) res))
5         (T (ra (cdr lst) (ra (car lst) res)))))
6
7 (defun rec-add (lst)
8   (ra lst 0))
```

1.7 Задание №7

Написать рекурсивную версию с именем `recnth` функции `nth`.

```
1 (defun rec-num-nth (n lst cur-ind)
2   (cond ((null lst) Nil)
3         ((= cur-ind n) (car lst))
4         (T (rec-num-nth n (cdr lst) (+ cur-ind 1)))))
5
6 (defun recnth (n lst)
7   (rec-num-nth n lst 0))
```

1.8 Задание №8

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

```
1 (defun allodd (lst)
2   (cond ((null lst) T)
3         ((evenp (car lst)) Nil)
4         (T (allodd (cdr lst)))))
```

1.9 Задание №9

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

```
1 (defun first-odd (lst)
2   (cond ((null lst) Nil)
3         ((and (numberp (car lst)) (oddp (car lst))) (car lst))
4         ((atom (car lst)) (first-odd (cdr lst)))
5         (T (or (first-odd (car lst)) (first-odd (cdr lst))))))
```

1.10 Задание №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.

```
1 (defun squares (lst)
2   (cond ((null lst) Nil)
3         (T (cons (* (car lst) (car lst)) (squares (cdr lst))))))
```