



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, искусственный интеллект и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Отчет по лабораторной работе №6.1
по курсу «Функциональное и логическое
программирование»
«Рекурсивные функции»

Студент группы **ИУ7-64Б**

(Подпись, дата)

Д.С. Чепиго

(И.О. Фамилия)

Преподаватели

(Подпись, дата)

Н.Б. Толпинская

(И.О. Фамилия)

(Подпись, дата)

Ю.В. Строганов

(И.О. Фамилия)

2023 г.

Содержание

1	Практические задания	3
1.1	Задание №1	3
1.2	Задание №2	3
1.3	Задание №3	3
1.4	Задание №4	4
1.5	Задание №5	5

1 Практические задания

Используя функционалы:

1.1 Задание №1

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

```
1 (defun my-reverse (lst &optional (buf-lst NIL))
2   (cond ((null lst) buf-lst)
3         (T (my-reverse (cdr lst) (cons (car lst) buf-lst)))))
```

1.2 Задание №2

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

```
1 (defun not-null-list (lst)
2   (cond ((null lst) NIL)
3         ((and (listp (car lst))
4               (not (null (caar lst)))) (car lst))
5         (T (not-null-list (cdr lst)))))
```

1.3 Задание №3

Напишите рекурсивную функцию, которая умножает на заданное число-аргумент все числа из заданного списка-аргумента, когда: все элементы списка – числа, элементы списка – любые объекты

```
1 (defun append-elem (lst elem &optional (before-lst Nil))
2   (cond ((and (null lst)
3               (null before-lst)) (cons elem NIL))
4         ((null lst)
5          (my-reverse (cons elem before-lst)))
6         (T
```

```

7         (append-elem (cdr lst) elem
8                     (cons (car lst) before-lst))))))
9
10 (defun all-numbers (num lst &optional (res-lst NIL))
11     (cond ((null lst) res-lst)
12           (T (all-numbers num (cdr lst)
13                          (append-elem res-lst (* (car lst) num))))))
14
15 (defun not-all-numbers (num lst &optional (res-lst NIL))
16     (cond ((null lst) res-lst)
17           ((numberp (car lst))
18            (not-all-numbers num (cdr lst)
19                             (append-elem res-lst (* num (car lst)))))
20           ((listp (car lst))
21            (not-all-numbers num (cdr lst)
22                             (append-elem res-lst
23                                           (not-all-numbers num (car lst)))))
24           (T (not-all-numbers num (cdr lst)
25                             (append-elem res-lst (car lst)))))

```

1.4 Задание №4

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 select-between (num1 num2 lst &optional (res-lst NIL))
2     (cond ((null lst) res-lst)
3           (< num1 (car lst) num2)
4            (select-between num1 num2 (cdr lst)
5                          (append-elem res-lst (car lst))))
6           (T (select-between num1 num2 (cdr lst) res-lst))))

```

1.5 Задание №5

Написать рекурсивную версию (с именем `rec-add`) вычисления суммы чисел заданного списка: одноуровневого смешанного, структурированного.

```
1 (defun rec-add-a (lst &optional (sum 0))
2   (cond ((null lst)
3         sum)
4         ((numberp (car lst))
5          (rec-add-a (cdr lst) (+ sum (car lst)))))
6   (T
7    (rec-add-a (cdr lst) sum))))
8
9 (print (rec-add-a '(1 v 4 0)))
10
11 (defun rec-add-b (lst &optional (sum 0))
12   (cond ((null lst)
13         sum)
14         ((numberp (car lst))
15          (rec-add-b (cdr lst) (+ sum (car lst)))))
16         ((listp (car lst))
17          (rec-add-b (cdr lst)
18                     (+ sum (rec-add-b (car lst)))))
19         (T
20          (rec-add-b (cdr lst) sum))))
```