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 № 6.1 по курсу «Функциональное и логическое программирование»

Тема Рекурсивные функции

Студент Виноградов А. О.

Группа ИУ7-66Б

Оценка (баллы)

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

Москва — 2023 г.

Введение

Цель работы: приобрести навыки организации рекурсии в Lisp.

Задачи работы: изучить способы организации хвостовой, дополняемой, множественной, взаимной рекурсии и рекурсии более высокого порядка в Lisp.

1 Практические задания

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

```
(defun my_reverse(lst)
  (add_to_res lst Nil)
)

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

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

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

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

а) все элементы списка — числа, б) элементы списка – любые объекты.

```
(defun f3_a (num lst)
  (defun f (num lst answ)
    (cond
        ((null (car lst)) answ)
        (t (f num (cdr lst) (cons (* (car lst) num) answ)))
    )
  )
  (f num lst ())
)
```

```

(defun f3_b (num lst)
  (defun f (num lst answ)
    (cond
        ((null (car lst)) answ)
        ((numberp (car lst)) (f num (cdr lst) (cons (* (car lst)
           num) answ)))
        (t (f num (cdr lst) (cons (car lst) answ)))
    )
  )
  (f num lst ())
)
```

1.4 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

```
(defun select-between (lst num1 num2)
  (defun f (lst num1 num2 res)
    (cond
        ((null (car lst)) res)
        ((or (< num1 (car lst) num2) (> num1 (car lst) num2)) (f
           (cdr lst) num1 num2 (cons (car lst) res)))
        (t (f (cdr lst) num1 num2 res))
    )
  )
  (f lst num1 num2 ())
)
```

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

```
(defun rec-add-a (lst)
  (defun f (lst sum)
    (cond
        ((null (car lst)) sum)
        ((numberp (car lst)) (f (cdr lst) (+ sum (car lst))))
        (t (f (cdr lst) sum))
    )
  )
  (f lst 0)
)
```

```
(defun rec-add-b (lst)
  (defun f (lst sum)
    (cond
        ((null (car lst)) sum)
        ((listp (car lst)) (f (cdr lst) (+ sum (f (car lst) 0))))
        (t (f (cdr lst) (+ sum (car lst))))
    )
  )
  (f lst 0)
)
```