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 №5 по дисциплине «Функциональное и логическое программирование»

Тема: Использование функционалов.
Студент: Карпова Е. О.
Группа: ИУ7-62Б
Оценка (баллы): _____
Преподаватели: Толпинская Н. Б., Строганов Ю. В.

Москва — 2023 г.

Оглавление

1. Практическая часть

1.1. Задание №1

Напишите функцию, которая уменьшает на 10 все числа из списка-аргумента этой функции, проходя по верхнему уровню списковых ячеек. (* Список смешанный структурированный)

Листинг 1.1 — Задание №1

```
[2]> (defun minus_10 (l)
        (mapcar #'(lambda (x) (if (numberp x) (- x 10) x)) l)
)
MINUS_10

[3]> (minus_10 '(20 (30 40) d "tgg" 50))
(10 (30 40) D "tgg" 40)
```

1.2. Задание №2

Написать функцию которая получает как аргумент список чисел, а возвращает список квадратов этих чисел в том же порядке.

Листинг 1.2 — Задание №2

```
[5]> (defun square_list (l)
        (mapcar #'(lambda (x) (if (numberp x) (* x x) x)) l)
)
SQUARE_LIST

[6]> (square_list '(2 3 (4 4) d f 6))
(4 9 (4 4) D F 36)
```

1.3. Задание №3

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

Листинг 1.3 — Задание №3

```
[7]> (defun mul_list (x mul)
        (mapcar #'(lambda (num) (* num mul)) x)
)
[8]> (mul_list '(1 2 3 4) 5) ; (5 10 15 20)
[11]> (defun mul_list (x mul)
        (mapcar #'(lambda (num) (if (numberp num) (* num mul) num)) x)
)
[12]> (mul_list '(1 2 3 (3 3) t "djd" 4) 5) ; (5 10 15 (3 3) T "djd" 20)
```

1.4. Задание №4

Написать функцию, которая по своему списку-аргументу lst определяет является ли он палиндромом (то есть равны ли lst и (reverse lst)), для одноуровнего смешанного списка.

Листинг 1.4 — Задание №4

```
(defun is_palindrome (x)
    (every #'eql x (reverse x))
)

(defun is_palindrome2 (lst)
        (reduce #'(lambda (x y) (and x y)) (mapcar #'eql lst
        (reverse lst)) :initial-value t)
)

(is_palindrome2 '(1 2 3 2 1)) ; T
(is_palindrome2 '(1 2 3)) ; NIL
(is_palindrome2 '()) ; T
```

1.5. Задание №5

Используя функционалы, написать предикат set-equal, который возвращает t, если два его множества-аргумента (одноуровневые списки) содержат одни и те же элементы, порядок которых не имеет значения.

Листинг 1.5 — Задание №5

```
(defun is_in_set(elem src_set)
        (not (null (member elem src_set))))
(defun is_subset(set1 set2)
    (reduce #'(lambda (x y) (and x y))
            (mapcar #'(lambda (x) (in_set x set2)) set1) :initial-value t))
(defun are_equal_sets(s1 s2)
    (and (is_subset s1 s2) (is_subset s2 s1)))

(are_equal_setsL '(1 3 4) '(1 4 3)) ; T
(are_equal_sets '() '()) ; T
(are_equal_sets '(1 2) '(1)) ; NIL
```

1.6. Задание №6

Напишите функцию, select-between, которая из списка-аргумента, содержащего только числа, выбирает только те, которые расположены между двумя указанными числами - границами-аргументами и возвращает их в виде списка (упорядоченного по возрастанию).

Листинг 1.6 — Задание №6

```
(defun select_between(l left right)
        (reduce #'(lambda (res elem)
            (if (< left elem right) (cons elem res) res))
                l :initial-value ()))

(defun select_between(l left right)
        (sort (reduce #'(lambda (res elem)
            (if (< left elem right) (cons elem res) res))
                l :initial-value ()) #'<))
```

1.7. Задание №7

Написать функцию, вычисляющую декартово произведение двух своих списков- аргументов. (Напомним, что A x B это множество всевозможных пар (a b), где a принадлежит A, b принадлежит B.)

Листинг 1.7 — Задание №7

```
(defun cartesian (l1 l2)
        (mapcan
                (lambda (x) (mapcar
                        (lambda (y) (cons x y))
                        l2
                ))
                l1
        )
)
CARTESIAN

(defun cartesian2 (l1 l2)
        (reduce #`append
        (mapcar
                (lambda (x) (mapcar
                        (lambda (y) (cons x y))
                        l2
                ))
                l1
        )
    :initial-value ())
)

(cartesian '(1 2 3) '(4 5 6))
((1 . 4) (1 . 5) (1 . 6) (2 . 4) (2 . 5) (2 . 6) (3 . 4) (3 . 5) (3 . 6))
```

1.8. Задание №8

Почему так реализовано reduce, в чем причина?

$(reduce \quad \#'+ \quad ()) \quad -> \quad 0,$

$(reduce \quad \#'* \quad ()) \quad -> \quad 1.$

Функционалы $+, *$ могут быть вызваны с 0 аргументов, при этом они будут возвращать нейтральные относительно операции значения: $+$ — 0, $*$ — 1.

: $initial - value$ не обязателен к указанию, но:

1. если нет : $initial - value$ и в аргументе > 1 элемента, то функция вызывается с первыми двумя элементами аргумента;
2. если нет : $initial - value$ и в аргументе 1 элемент, то возвращается значение этого элемента и функция не вызывается;
3. если есть : $initial - value$ и аргумент пуст, то возвращается : $initial - value$ и функция не вызывается;
4. если нет : $initial - value$ и аргумент пуст, то функция вызывается с 0 аргументов.

1.9. Задание №9

Пусть $list - of - list$ список, состоящий из списков. Написать функцию, которая вычисляет сумму длин всех элементов $list - of - list$ (количество атомов), т.е. например для аргумента $((1 \quad 2) \quad (3 \quad 4)) \quad -> \quad 4.$

Листинг 1.8 — Задание №9

```
(defun list_of_list_len (l)
(reduce
  (lambda (res elem) (+ res (length elem)))
  l :initial-value 0)
)

(list_of_list_len '((1 3 4) (2 3) (1)))
6
(list_of_list_len '())
0
```