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 курсу «Функциональное и логическое программирование»

Тема Использование функционалов

Студент Динь Вьет Ань

Группа ИУ7И-64Б

Оценка (баллы) _____

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

Практические задания

Задание 1

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

```
(defun minus-ten (lst)
(
    mapcar #'(lambda (el)
        ( cond
                ((numberp el) (- el 10)
                (T el))
        ) lst
    )
)
```

Задание 2

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

```
(defun sqr-lst (lst)
(
    mapcar #'(lambda (el)
        (* el el)) lst
    )
)
```

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.

а)

```
(defun mul-lst (lst num)
(
    mapcar #'(lambda (el)
        (* el num)) lst
    )
)
```

б)

```
(defun mul-lst (lst num) (
    mapcar #'(lambda (el) (
        (cond ((numberp el) (* el num)
            (T el))
        ) lst
    )
)
```

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.

```
(defun list_without_last (lst res)
    (cond
        ((null (cdr lst)) res)
        (t (list_without_last (cdr lst) (cons (car lst) res)))
))

(defun is_palindrome (lst)
    (cond
        ((null (cdr lst)) t)
        ((eql (car lst) (car (last lst)))
            (is_palindrome (list_without_last (cdr lst) ())))
))
```

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.

```
(defun set-equal (set1 set2)
    (cond
        ((null set1) (null set2))
        ((null set2) Nil)
        (t (set-equal (cdr set1) (remove (car set1) set2)))
))
```

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 (+ 2 балла)).

```
(defun select-between (lst left right) (
    cond ((< right left) Nil)
    (T (sort (reduce #'(lambda (result el)
              (cond ((and (> el left) (> right el))
                 (cons el result))
            (T result))
        )lst :initial-value '()) #'<
))))
```

Задание 7

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

```
(defun decart (lst1 lst2) (
    mapcan #'(lambda (x)
        (mapcar #'(lambda (y)
            (list x y)) lst2)) lst1
))
```

Задание 8

Почему так реализовано reduce, в чем причина?

```
(reduce #'+ ()) -> 0
(reduce #'* ()) -> 1
```

Если список пуст, а начальное значение не задано, то вызывается функция без агрументов, в reduce возвращает то, что вернёт функция. Функция сложения без аргументов возвращает 0, а функцуия умножения возвращает 1.

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 2) (3 4)) -> 4.

```
(defun len-list-of-lists (lst) (
    apply #'+ (
        mapcar #'(lambda (el)
            (cond ((listp el) (len-list-of-lists el))
            (T 1))
        ) lst
)))
```