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
по дисциплине
"Функциональное и логическое программирование"

Тема Использование функционалов

Студент Малышев И. А.

Группа ИУ7-61Б

Оценка (баллы) _____

Преподаватель: Толпинская Н. Б.

Москва — 2022 г.

Практические задания

1. Напишите функцию, которая уменьшает на 10 все числа из списка-аргумента этой функции.

Листинг 1: Решение задания №1

```
(defun lst-minus-10 (lst)
  (mapcar #'(lambda (x) (- x 10)) lst))
```

2. 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 2: Решение задания №2

```
; a)
(defun mult-all-numbers (mult lst)
  (mapcar #'(lambda (el) (* el mult)) lst))

; b)
(defun compl-mult-all-numbers (mult lst)
  (mapcar #'(lambda (el)
        (cond ((listp el) (compl-mult-all-numbers mult el))
              ((numberp el) (* el mult))
              (t el)))
      lst))
```

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

Листинг 3: Решение задания №3

```
(defun to-pairs (lst1 lst2)
  (mapcar #'equal lst1 lst2))

(defun is-same (lst1 lst2)
```

```
5   (every #'identity (to-pairs lst1 lst2)))
6
7 (defun is-palyndrome (lst)
8   (is-same lst (reverse lst)))
```

4. Написать предикат set-equal, который возвращает t, если два его множества-аргумента содержат одни и те же элементы, порядок которых не имеет значения.

Листинг 4: Решение задания №4

```
1 (defun comp (lst1 lst2)
2   (every #'identity (mapcar #'(lambda (elem)
3                 (if (find-if #'(lambda (x) (equal elem x)) lst2) T))
4             lst1)))
5
6 (defun set-equal (lst1 lst2) (if (= (length lst1) (length lst2)) (comp
     lst1 lst2)))
```

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

Листинг 5: Решение задания №5

```
1 (defun sqr-lst (lst)
2   (mapcar #'(lambda (x) (* x x)) lst))
```

6. 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 (упорядоченного по возрастанию списка чисел (+ 2 балла)).

Листинг 6: Решение задания №6

```
1 (defun betweenp (el b1 b2)
2   (< (* (- el b1) (- el b2)) 0))
3
4 (defun select-between (lst b1 b2)
5   (mapcan #'(lambda (elem) (if (betweenp elem b1 b2) (list elem))) lst))
```

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, принадлежит B.)

Листинг 7: Решение задания №7

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

8. Почему так реализовано reduce, в чем причина?

(reduce #'+ 0) -> 0
(reduce #'+ ()) -> 0

Поведение в данном примере обусловлено работой функции `+`: при нулевом количестве аргументов возвращает значение 0. Если подать на вход `reduce` функцию, которая не может обработать 0 аргументов (например, `cons`), то вызов `reduce` с пустым списком в качестве второго аргумента вернет ошибку (`invalid number of arguments: 0`). При этом, если подано более одного аргумента, то `reduce` выполняет следующие действия:

1. сохраняет первый элемент списка в область памяти (для определенности назовем ее `acc`);
2. для всех остальных элементов списка выполняет переданную в качестве первого аргумента функцию, подавая на вход 2 аргумента (`acc` и очередной элемент списка) и сохраняя результат в `acc`.

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

Листинг 8: Решение задания №9

```
(defun sum-lens (list-of-lists)
  (reduce #'(lambda (acc lst) (+ acc (length lst)))
      list-of-lists :initial-value 0))
```