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 № 5 по дисциплине "Функциональное и логическое программирование"

Студент Кузнецова А. В.

Группа ИУ7-61Б

Оценка (баллы)

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

Москва — 2023 г.

1 Практические задания

Используя функционалы:

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)

```
(defun minus10 (lst)
    (mapcar #'(lambda (x)
        (cond ((numberp x) (- x 10))
            ((atom x) x)
            (T (minus10 x)))) lst))
```

1.2 Задание №2

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

```
(defun squares (lst)
    (mapcar #'(lambda (x) (* x x)) lst))
```

1.3 Задание №3

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

а) все элементы списка — числа,

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

б) элементы списка — любые объекты.

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

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.

```
    (defun is-palindrome (lst)
        (equal lst (reverse lst)))
```

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.

```
    (defun is-in (el st)
        (reduce #'(lambda (x y) (or x y))
            (mapcar #'(lambda (x) (equal x el)) st)
               :initial-value Nil))

    (defun is-subset (st1 st2)
        (reduce #'(lambda (x y) (and x y))
            (mapcar #'(lambda (x) (is-in x st2)) st1)
               :initial-value T))

    (defun set-equal (st1 st2)
        (and (is-subset st1 st2) (is-subset st2 st1)))
```

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 (+ 2 балла)).

```
1  (defun select-between (lst begin end)
2      (sort (remove-if-not #'(lambda (x) (< begin x end)) lst)
          '<))
```

1.7 Задание №7

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

```
1  (defun cartesian (lst1 lst2)
2      (mapcan #'(lambda (el1)
3          (mapcar #'(lambda (el2) (list el1 el2)) lst2)) lst1))
```

1.8 Задание №8

Почему так реализовано reduce, в чем причина?

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

Функционал reduce выполняет следующее преобразование исходного списка L=>(e1 e2 ... en) с использованием начального значения A и бинарной операции-функции F: (reduce F L A) = (F(...(F(F A e1) e2))...en). Можно сказать, что значение A логически добавляется в начало списка L.

- Если список содержит ровно один элемент и начальное значение не задано, то этот элемент возвращается, а функция не вызывается.

- Если список пуст и задано начальное значение, то возвращается начальное значение, а функция не вызывается.

- Если список пуст и начальное значение не задано, то функция вызывается без аргументов, и reduce возвращает то, что вернет функция. Это единственный случай, когда функция вызывается с другим количеством аргументов, кроме двух.

(reduce #'+ ()): список пуст, начальное значение не указано, следовательно, функция + вызывается без аргументов, и reduce возвращает то, что вернет функция +. (+) => 0, поэтому и (reduce #'+ ()) => 0.

С умножением аналогично.

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 2) (3 4)) -> 4.

```
1  (defun get-length (lst)
2    (reduce #'+ (mapcar #'(lambda (x) (cond ((atom x) 1)
3                                            (T (get-length
                                                 x)))) lst)))
```