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» (ИУ7)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.04 Программная инженерия**

О Т Ч Е Т

по лабораторной работе № 9

Дисциплина: Функциональное и логическое программирование

Студент	ИУ7-62Б		Е.В. Брянская
	(Группа)	(Подпись, дата)	(И.О. Фамилия)
Преподаватель			Н.Б.Толпинская
		(Подпись, дата)	(И.О. Фамилия)

Москва, 2021

1. Написать функцию, которая выбирает из заданного списка только те числа, которые больше 1 и меньше 10 (Вариант: между двумя заданными границами)

```
(defun find-elements (lst left right)
        (remove-if #'(lambda (x) (null x))
                        (mapcar #'(lambda (x)
                                        (if (< left x right)
                                                x)) lst)))


(defun select-between (lst b1 b2)
        (cond ((null lst) nil)
            ((not (and (numberp b1) (numberp b2)))(and (print "ERROR: wrong format of borders") nil))
            ((= b1 b2)(and (print "ERROR: wrong format of borders (equal)") nil))
            ((> b1 b2)(find-elements lst b2 b1))
            ((> b2 b1)(find-elements lst b1 b2))))
```

2. 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)

```
(defun decart (lstx lsty)
        (mapcan #'(lambda (x)
                        (mapcar #'(lambda (y)
                                        (list x y)) lsty)) lstx))
```

3. Почему так реализовано reduce, в чём причина?

(reduce #’+ ()) -> 0

(reduce #’* ()) -> 1

Дело в том, что у reduce есть особый параметр – initial-value, который помещается перед последовательностью и затем применяется функция. Его можно задать, используя:

(reduce #’+ () :initial-value 100) -> 100

По умолчанию, для сложения оно равно 0, для умножения 1.

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

1 вариант

```
(defun count-length (lst len)
        (cond ((null lst) len)
            ((atom (car lst)) (count-length (cdr lst) (+ len 1)))
            (t (count-length (cdr lst) (count-length (car lst) len)))))


(defun count-list-length (lst)
        (count-length lst 0))
```

2 вариант

```
(defun count-list-length-2 (lst)
        (reduce #'(lambda (x y)
                        (+ x (cond ((null y) 1)
                                ((listp y) (count-list-length-2 y))
                                (t 1)))
                        )
                lst
                :initial-value 0))
```

5. Используя рекурсию, написать функцию, которая по исходному списку строит список квадратов чисел смешанного структурированного списка.

```
(defun find-square (lst)
        (cond ((null lst) t)
                ((listp (car lst))(find-square (car lst)))
                ((numberp (car lst))(and (rplaca lst (* (car lst) (car lst)))
                                        (find-square (cdr lst))))
               (t (find-square (cdr lst))))
        lst)
```

Вопросы

1. Классификация рекурсивных функций

Рекурсия:

1. Простая рекурсия
2. Рекурсия первого порядка
3. Взаимная рекурсия

Рекурсия:

1. Хвостовая рекурсия
2. Дополняемая рекурсия
    a. Cons-дополняемая рекурсия
    b. Когда дополнительная функция – способ прервать рекурсию
    c. Дополнительная функция комбинирует два рекурсивных вызова