



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 №6 по курсу

«Функциональное и логическое программирование»

Тема Использование функционалов.

Студент Сироткина П.Ю.

Группа ИУ7-66Б

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

Москва — 2022 г.

Практические задания

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

```
1 (defun dec_10(lst)
2   (mapcar #'(lambda (elem)
3             (cond ((numberp elem) (- elem 10)) (T elem))) lst))
```

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

1. Все элементы списка – числа;
2. Элементы списка – любые объекты.

```
1 (defun mul_num(lst multiplier)
2   (mapcar #'(lambda(elem) (* elem multiplier)) lst))
3
4 (defun mul(lst multiplier)
5   (mapcar #'(lambda (elem)
6             (cond ((listp elem) (mul (cdr elem) multiplier))
7                   ((numberp elem) (* elem multiplier))
8                   (T elem))) lst))
```

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

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

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

```
1 (defun make_squares(lst) (mapcar #'(lambda (elem) (* elem elem)) lst))
```

6. Написать 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)).

```
1 (defun select_between(lst left right)
2   (sort (reduce #'(lambda (res_lst elem)
3     (if (< left elem right) (cons elem res_lst) res_lst))
4     lst :initial-value ()) #'<))
```

7. Написать функцию, вычисляющую декартово произведение двух своих списков-аргументов.

```
1 (defun decart(lst_x lst_y)
2   (mapcan #'(lambda(x) (mapcar #'(lambda(y) (list x y)) lst_y)) lst_x))
```

8. Почему так реализовано `reduce`, в чем причина?

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

Начальный результат работы функционала **reduce** - т.н. нейтральный. Для операции сумма нейтральным элементом является 0.

9. Пусть `list-of-list` список, состоящий из списков. Написать функцию, которая вычисляет сумму длин всех элементов `list-of-list`.

```
1 (defun sum_lengths lol)
2   (reduce #'(lambda(cur_sum lst) (+ cur_sum (length lst))) (cons 0 lol)))
```