Министерство науки и высшего образования Российской ФедерацииФедеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет имениН.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 курсу «Функциональное и логическое программирование»

Тема Использование функционалов

Студент Климов И.С.

Группа ИУ7-62Б

Оценка (баллы) _____

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

Москва — 2022 г.

Задание 1

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

Решение

```
(defun apply_opr_to_numbers (lst opr num)
  (mapcar #'(lambda (x) (if (numberp x) (eval `(,opr ,x ,num)) x)) lst))

(defun reduce_by_10 (lst)
  (apply_opr_to_numbers lst '- 10))
```

Задание 2

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

a) все элементы списка – числа,

b) элементы списка – любые объекты.

Решение

```
(defun mul_by_number_v1 (lst num)
  (mapcar #'(lambda (x) (* x num)) lst))

(defun mul_by_number_v2 (lst num)
  (apply_opr_to_numbers lst '* num))
```

Задание 3

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

Решение

```
(defun my_reverse (lst)
  (let ((reverse_list NIL))
    (mapcar #'(lambda (x) (setf reverse_list
                                (append (list x) reverse_list))) lst)
    reverse_list))
(defun palindrome (lst)
  (cond ((null (listp lst)) NIL)
      (T (equal lst (my_reverse lst)))))
```

Задание 4

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

Решение

```
(defun and_list (lst)
  (eval `(and ,@lst)))

(defun or_list (lst)
  (eval `(or ,@lst)))

(defun my_member (element lst)
  (or_list (mapcar #'(lambda (x) (equal x element)) lst)))

(defun my_subsetp (lst1 lst2)
  (and_list (mapcar #'(lambda (x) (my_member x lst2)) lst1)))

(defun set_equal (lst1 lst2)
  (and (listp lst1) (listp lst2)
       (my_subsetp lst1 lst2) (my_subsetp lst2 lst1)))
```

Задание 5

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

Решение

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

Задание 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)).

Решение

```
(defun my_max (lst &optional (max NIL))
  (mapcar #'(lambda (x) (if (or (null max) (> x max))
                      (setq max x))) lst)
  max)

(defun my_length (lst)
  (apply #'+ (mapcar #'(lambda (x) (declare (ignore x)) 1) lst)))

(defun butfirst (lst num)
  (car (remove NIL (maplist #'(lambda (x)
                        (if (= (my_length x) (- (my_length lst) num))
                            x NIL)) lst))))

(defun my_remove (x lst)
  (cond ((or (null x) (null lst)) NIL)
      ((null (my_member x lst)) lst)
      (T (append (butlast lst (- (my_length lst) (position x lst)))
                (butfirst lst (+ (position x lst) 1))))))
```

```
(defun my_sort_help (lst result copied_lst)
  (mapcar #'(lambda (x)
            (declare (ignore x))
            (setq result (cons (my_max copied_lst) result))
            (setq copied_lst (my_remove (my_max copied_lst) copied_lst))) lst)
  result)

(defun my_sort (lst)
  (my_sort_help lst () lst))

(defun select_between_help (lst left right)
  (my_sort (remove NIL  (mapcar #'(lambda (num)
                   (if (and (> num left) (< num right)) num)) lst))))

(defun select_between (lst left right)
  (cond ((> left right) (select_between_help lst right left))
      (T (select_between_help lst left right))))
```

Задание 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.)

Решение

```
(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) -> 0

(reduce #'+ ()) -> 0

Решение

Reduce применяет функцию к списку аргументов каскадным образом. То есть сначала функция применяется к первым двум аргументам, затем к получившемуся результату и третьему аргументу и т.д.

Первое выражение выдаст ошибку, так как 0 является атомом, а не списком. Второе выражение вернет 0, так как функция + может работать при отсутствии аргументов и возвращает при этом значение 0.

Задание 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.

Решение

```
(defun len_list_of_list (lst)
  (apply #'+ (mapcar #'length lst)))
```