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 курсу: "Функциональное и Логическое программирование"

Тема Использование управляющих структур, работа со списками.

Группа ИУ7-63Б (ИУ7и-67б)

Студент Тэмуужин Я.

Преподаватель Толпинская Н.Б.

Преподаватель Строганов Ю. В.

Москва — 2022 г.

Задание 1. Написать функцию, которая по своему списку-аргументу lst определяет является ли он палиндромом (то есть равны ли lst и (reverse lst)).

```
(defun is-pali-ins (tmp lst)
  (cond ((null lst) nil)
        ((or (equal tmp lst) (equal tmp (cdr lst))) T)
        (T (is-p2-ins (cons (car lst) tmp) (cdr lst) )) ))

(defun is-pali (lst) (is-p2-ins () lst))

(defun is-pali-2 (lst)
  (if (equal (reverse lst) lst) T))
```

Задание 2. Написать предикат set-equal, который возвращает t, если два его множества-аргумента содержат одни и те же элементы, порядок которых не имеет значения.

```
(defun my-member (el set2)
  (cond ((null set2) nil)
        ((equal el (car set2)) T)
        (T (my-member el (cdr set2)))))

(defun my-subsetp (set1 set2)
  (cond ((my-member (car set1) set2)
         (if (not (null (cdr set1)))
             (my-subsetp (cdr set1) set2) T)) ))

(defun set-equal (set1 set2)
  (and (my-subsetp set1 set2)
       (my-subsetp set2 set1)))

(defun set-equal-2 (lst1 lst2)
  (and (subsetp lst2 lst1) (subsetp lst1 lst2)))
```

Задание 3. Напишите свои необходимые функции, которые обрабатывают таблицу из 4-х точечных пар: (страна . столица), и возвращают по стране - столицу, а по столице — страну.

```
(defun get-country (table capital)
  (cond ((null table) Nil)
        ((eq capital (cdar table)) (caar table))
        (T (get-country (cdr table) capital))   ))

(defun get-capital (table country)
  (cond ((null table) Nil)
        ((eq country (caar table)) (cdar table))
        (T (get-capital (cdr table) country))   ))
```

Задание 4. Напишите функцию swap-first-last, которая переставляет в списке-аргументе первый и последний элементы.

```
(defun f-no-l1 (lst)
  (reverse (cdr (reverse lst))))

(defun swap-first-last (lst)
  (append (last lst) (cdr (f-no-l1 lst)) (cons (car lst) Nil)))
```

Задание 5. Напишите функцию swap-two-ellement, которая переставляет в списке- аргументе два указанных своими порядковыми номерами элемента в этом списке.

```
(defun swap-two-ellement1 (lst n1 n2)
  (let ((el1 (nth n1 lst))
        (el2 (nth n2 lst)))
   (setf (nth n1 lst) el2
         (nth n2 lst) el1)
    lst))

(defun mynthcdr (n lst)
  (cond ((null lst) Nil)
        ((eq 0 n) lst)
        (T (setf n (- n 1))
           (mynthcdr n (cdr lst))) ))

(defun swap-two-ellement2 (lst n1 n2)
  (let ((el1 (car (mynthcdr n1 lst)))
        (el2 (car (mynthcdr n2 lst))))
   (setf (car (mynthcdr n1 lst)) el2
         (car (mynthcdr n2 lst)) el1)
    lst))
```

Задание 6. Напишите две функции, swap-to-left и swap-to-right, которые производят одну круговую перестановку в списке-аргументе влево и вправо, соответственно.

```
(defun f-no-l1 (lst)
  (reverse (cdr (reverse lst))))

;; (1 2 3 4 5) -> (2 3 4 5 1)
(defun swap-to-left (lst)
  (append (cdr lst) (cons (car lst) Nil)))
;; (1 2 3 4 5) -> (5 1 2 3 4)
(defun swap-to-right (lst)
  (cons (car (last lst)) (f-no-l1 lst)))
```

Задание 7. Напишите функцию, которая добавляет к множеству двухэлементных списков новый двухэлементный список, если его там нет.

```
(defun add-lst-to-set-ins (lset lst tmp)
  (cond ((null tmp) (append lset (list lst)))
        ((equal (car tmp) lst) lset)
        (T (add-lst-to-set-ins lset lst (cdr tmp)))))
(defun add-lst-to-set (lset lst)
  (add-lst-to-set-ins lset lst lset))
```

Задание 8. Напишите функцию, которая умножает на заданное число-аргумент первый числовой элемент списка из заданного 3-х элементного списка-аргумента, когда

а) все элементы списка – числа, ;; 10 '(1 2 3) -> (10 2 3)

б) элементы списка – любые объекты. ;; 10 '((r 3) 4 sfsd) -> ((r 3) 40 sfsd)

```
(defun mult-a (x lst)
  (cond ((/= (length lst) 3) 'length!=3)
        (T (setf (car lst) (* x (car lst)))
           lst) ))

(defun mult-b (x lst)
  (cond ((/= (length lst) 3) 'length!=3)
        ((numberp (car lst))
         (setf (car lst) (* x (car lst))) lst)
        ((numberp (cadr lst))
         (setf (cadr lst) (* x (cadr lst))) lst)
        ((numberp (caddr lst))
         (setf (caddr lst) (* x (caddr lst))) lst) ))

(defun mult-b2 (x lst plst)
  (cond ((null lst) plst)
        ((numberp (car lst))
         (setf (car lst) (* x (car lst))) plst)
        (T (mult-b2 x (cdr lst) plst) )))
```

Задание 9. Напишите функцию, select-between, которая из списка-аргумента из 5 чисел выбирает только те, которые расположены между двумя указанными границами-аргументами и возвращает их в виде списка (упорядоченного по возрастанию списка чисел (+ 2 балла)).

```
(defun add-to-end (x lst p)
  (cond ((null (car lst)) (setf (car lst) x) p)
        ((null (cdr lst)) (setf (cdr lst) (cons x Nil)) p)
        (T (add-to-end x (cdr lst) p)) ))
(defun is-between (x b1 b2)
  (or (and (> x b1) (< x b2))
      (and (< x b1) (> x b2))))

;; (setf res (cons nil nil))
(defun select-between (lst b1 b2 res)
  (cond ((null lst) res)
        ((is-between (car lst) b1 b2)
         (add-to-end (car lst) res res)
         (select-between (cdr lst) b1 b2 res))
        (T (select-between (cdr lst) b1 b2 res))))
```