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»

Студент Маслова Марина Дмитриевна

Группа ИУ7-63Б

Оценка (баллы) _____

Преподаватель Толпинская Наталья Борисовна

Преподаватель Строганов Юрий Владимирович

2022 г.

1 Практические задания

1.1 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-palindrome (lst)
    (equal lst (reverse lst)))
```

1.2 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 set-equal1 (set1 set2)
    (and (subsetp set1 set2) (subsetp set2 set1)))
```

1.3 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-capital (table country)
    (cond ((null table) Nil)
          ((equal (caar table) country) (cdar table))
          (T (get-capital (cdr table) country))))
```

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

1.4 Задание №4

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

```
(defun swap-first-last (lst)
    (cond ((null (cdr lst)) lst)
          (T (cons (car (last lst))
                   (reverse (cons (car lst) (cdr (reverse (cdr lst)))))))))
```

1.5 Задание №5

Напишите функцию swap-two-element, которая переставляет в списке-аргументе два указанных своими порядковыми номерами элемента в этой списке.

```
(defun ste (lst num1 num2 cur-ind first-el res)
    (cond ((null lst) (reverse res))
          ((= cur-ind num1)
              (ste (cdr lst) num1 num2 (+ cur-ind 1) (car lst)
                   (cons (nth (- num2 num1) lst) res)))
          ((= cur-ind num2)
              (ste (cdr lst) num1 num2 (+ cur-ind 1) first-el
                   (cons first-el res)))
          (T (ste (cdr lst) num1 num2 (+ cur-ind 1) first-el
                  (cons (car lst) res)))))

(defun swap-two-element (lst num1 num2)
    (cond
     ((or (>= num1 (length lst))
          (>= num2 (length lst))
          (< num1 0)
          (< num2 0))
      Nil)
     ((< num1 num2) (ste lst num1 num2 0 Nil ()))
     (T (ste lst num2 num1 0 Nil ()))))
```

1.6 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 swap-to-left (lst)
     (append (cdr lst) (cons (car lst) Nil)))
```

```
(defun swap-to-right (lst)
     (cons (car (last lst)) (reverse (cdr (reverse lst)))))
```

1.7 Задание №7

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

```
(defun adel (list-set lst is-in res)
    (cond ((null list-set) (cons (list lst) res))
          ; тут нужна проверка is-in еще вместе с null
          (T (add-two-el-list (cdr list-set) lst (or is-in (equal (car
              list-set)
          ) lst (cons (car list-set) res))))

(defun add-two-el-list (list-set lst)
    (adel list-set lst Nil ()))
```

1.8 Задание №8

Напишите функцию, которая умножает на заданное число-аргумент первый числовой элемент списка из заданного 3-хэлементного списка-аргумента когда:

1. все элементы списка — числа,

```
(defun mul-first-nums (lst num)
    (and lst (setf (car lst) (* (car lst) num)) lst))
```

2. элементы списка — любые объекты.

```
(defun mul-first-num (lst num)
    (cond ((null lst) lst)
          ((numberp (car lst)) (cons (* (car lst) num) (cdr lst)))
          (T (cons (car lst) (mul-first-num (cdr lst) num)))))
```

1.9 Задание №9

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

```
(defun select-between (lst begin end)
    (cond ((null lst) Nil)
          ((not (< begin (car lst) end))
                (select-between (cdr lst) begin end))
          (T (cons (car lst) (select-between (cdr lst) begin end)))))

(defun sort-select-between (lst begin end)
    (sort (select-between lst begin end) #'<))
```

2 Теоретические вопросы

2.1 Структуроразрушающие и не разрушающие структуру списка функции

Структуроразрушающие функции — функции, после использования которых теряется возможность работы изначальными списками. Эти функции изменяют сам объект. Чаще всего такие функции начинаются с префикса `n` (например, `nconc`, `nreverse` и т. п.).

Функции, не разрушающие структуру списка — функции, после использования которых сохраняется возможность работы с изначальными списками. Эти функции не изменяют сам объект, а создают копии (например, `append`, `reverse`, `length` и т. п.).

2.2 Отличие в работе функций cons, list, append, nconc и в их результате

`conc` создает списковую ячейку и расставляет указатели (car-указатель — на первый аргумент, cdr-указатель — на второй аргумент), если второй аргумент — список, то возвращает список, если нет — точечную пару.

`list` создает столько списковых ячеек, сколько аргументов, возвращет список.

`append` создает копии всех элементов кроме последнего и расставляет cdr-указатели последних элементов копий на следующие копии/аргумент.

`nconc` аналогична `append` только работает не с копиями, а с исходными списками.