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 дисциплине
"Функциональное и логическое программирование"

Тема Использование управляющих структур, работа со списками

Студент Малышев И. А.

Группа ИУ7-61Б

Оценка (баллы) _____

Преподаватель: Толпинская Н. Б.

Москва — 2022 г.

Теоретические вопросы

1. Структуроразрушающие и не разрушающие структуру списка функции

Не разрушающие структуру списка функции. Данные функции не меняют сам объект-аргумент, а создают копию. К таким функциям относятся: `append`, `reverse`, `last`, `nth`, `nthcdr`, `length`, `remove`, `subst` и прочие.

Структуроразрушающие функции. Данные функции меняют сам объект-аргумент, невозможно вернуться к исходному списку. Чаще всего такие функции начинаются с префикса `n-`. К такми функция относятся: `nreverse`, `nconc`, `nsubst` и прочие.

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

Функция `cons` — чисто математическая, конструирует списковую ячейку, которая может вовсе и не быть списком. Является списком только в том случае, если вторым аргументом передан список.

Функция `list` — форма, принимает произвольное количество аргументов и конструирует из них список. Результат — всегда список. При нуле аргументов возвращает пустой список.

Функция `append` — форма, принимает на вход произвольное количество аргументов и для всех аргументов, кроме последнего, создает копию, ссылая при этом последний элемент каждого списка-аргумента на первый элемент следующего по порядку списка-аргумента. Копирование для последнего не делается в целях эффективности.

Практические задания

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

Листинг 1: Решение задания №1

```
(defun is-palindrome (lst)
  (equal lst (reverse lst)))
```

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

Листинг 2: Решение задания №2

```
(defun set-equal (set1 set2)
  (if (= (length set1) (length set2))
    (and (subsetp set1 set2) (subsetp set2 set1))
    Nil))
```

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

Листинг 3: Решение задания №3

```
(defun get-capital (table country)
  (cdr (assoc country table)))

(defun get-country (table capital)
  (car (rassoc capital table)))
```

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

Листинг 4: Решение задания №4

```
(defun swap-first-last (lst)
  (let ((el1 (car lst)) (last-el (car (reverse lst))))
    (reverse (cons el1 (cdr (reverse (cons last-el (cdr lst))))))))
```

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

Листинг 5: Решение задания №5

```
(defun swap-two-ellement (n1 n2 lst)
  (let ((len (length lst)) (lst-copy (copy-list lst)))
    (and (< n1 len) (< n2 len)
    (let ((el1 (nth n1 lst)) (el2 (nth n2 lst)))
      (setf (nth n1 lst-copy) el2)
      (setf (nth n2 lst-copy) el1)
      lst-copy))))
```

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

Листинг 6: Решение задания №6

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

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

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

Листинг 7: Решение задания №7

```
(defun is-double-in-set (lst double)
  (cond ((null lst) nil)
        ((and (eql (caar lst) (car double)) (eql (cadar lst) (cadr
           double))) t)
        (t (is-double-in-set (cdr lst) double))))

(defun inner-append-double-lst (lst double)
  (cond ((null (cdr lst)) (cons (car lst) (cons double nil)))
        (t (cons (car lst) (inner-append-double-lst (cdr lst) double)))))

(defun append-double-lst (lst double)
  (if (is-double-in-set lst double)
    lst
    (inner-append-double-lst lst double)))
```

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,
б) элементы списка – любые объекты.

Листинг 8: Решение задания №8

```
; a)
(defun first-num-mul (lst mul)
  (cond ((null lst) nil)
        (t (cons (* (car lst) mul) (cdr lst)))))

; b)
(defun first-num-mul (lst mul)
  (cond ((null lst) nil)
        ((numberp (car lst)) (cons (* (car lst) mul) (cdr lst)))
        ((listp (car lst)) (cons (first-num-mul (car lst) mul))
           (first-num-mul (cdr lst) mul))
        (t (cons (car lst) (first-num-mul (cdr lst) mul)))))
```

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)).

Листинг 9: Решение задания №9

```
(defun betweenp (el b1 b2)
  (< (* (- el b1) (- el b2)) 0))

(defun select-between (lst b1 b2)
  (cond ((null lst) nil)
      ((betweenp (car lst) b1 b2) (cons (car lst) (select-between (cdr
         lst) b1 b2)))
      (T (select-between (cdr lst) b1 b2))))
```