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 №4 по дисциплине «Функциональное и логическое программирование»

Тема Использование урпавляющих структур, работа со списками

Студент Котляров Н.А.

Группа ИУ7-61Б

Оценка (баллы)

Преподаватель Толпинская Н. Б. Строганов Ю. В.

Москва — 2023 г.

Задание 1

Чем принципиально отличаются функции cons, list, append?

- cons — принимает 2 аргумента. Если второй аргумент является списком, создает 1 списковую ячейку, значением которой является первый аргумент, а список – ссылкой на второй аргумент (список). Если второй аргумент является атомом, создает точечную пару.
- list — принимает переменное количество аргументов. Создает столько списковых ячеек, сколько принято аргументов. Значением каждой списковой ячейки является значение принятых аргументов.
- append — принимает переменное количество аргументов. все аргументы, кроме последнего, должны быть списками. Возвращает новый список, содержащий конкатенацию копий. Списки остаются без изменений, копируется структура каждого списка, кроме последнего: он не копируется, а становится cdr последней точечной пары или возвращается напрямую, если нет предшествующих непустых списков.

Пусть (setf lst1 '(a b c)) (setf lst2 '(d e)).
Каковы результаты вычисления следующих выражений?

- (cons lstl lst2) — ((A B C) D E)
- (list lst1 lst2) — ((A B C) (D E))
- (append lst1 lst2) — (A B C D E)

Задание 2

- (reverse '(a b c)) — (C B A)
- (reverse '(a b (c (d)))) — ((C (D)) B A)
- (reverse '(a)) — (A)
- (reverse ()) — NIL
- (reverse '((a b c))) — ((A B C))
- (last '(a b c)) — (C)
- (last '(a)) — (A)
- (last '((a b c))) — ((A B C))
- (last '(a b (c))) — ((C))
- (last ()) — NIL

Задание 3

Написать, по крайней мере, два варианта функции, которая возвращает последний элемент своего списка-аргумента.

```
1 (defun last_el (lst)
2     (car (last lst))
3 )
```

```
1 (defun last_el (lst)
2     (car (reverse lst))
3 )
```

Задание 4

Написать, по крайней мере, два варианта функции, которая возвращает свой список аргумент без последнего элемента.

```
(defun without_last_el (lst)
    (
        reverse (cdr (reverse lst))
    )
)
```

```
(defun without_last_el (lst)
    (
        cond
        (
            (cdr lst)
            (cons (car lst) (without_last_el (cdr lst)))
        )
        (
            T
            Nil
        )
    )
)
```

Задание 5

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

```
(defun swap_first_last (lst)
    (
        append
        (last lst)
        (reverse (cdr (reverse (cdr lst))))
        (cons (car lst) Nil)
    )
)
```

Задание 6

Написать простой вариант игры в кости, в котором бросаются две правильные кости. Если сумма выпавших очков равна 7 или 11 — выигрыш, если выпало (1,1) или (6,6) — игрок имеет право снова бросить кости, во всех остальных случаях ход переходит ко второму игроку, но запоминается сумма выпавших очков. Если второй игрок не выигрывает абсолютно, то выигрывает тот игрок, у которого больше очков. Результат игры и значения выпавших костей выводить на экран с помощью функции print.

```
(defvar rs (make-random-state T))
(defvar thr1)
(defvar thr2)

(defun bones_throw ()
    (
        list
        (+ (random 6 rs) 1)
        (+ (random 6 rs) 1)
    )
)

(defun win_check (lst)
    (
        or
        (
            =
            (+ (car lst) (cadr lst))
            7
        )
        (
            =
            (+ (car lst) (cadr lst))
            11
        )
    )
)


```

```
(defun pass_check (lst)
    (
        or
        (
            and
            (= (car lst) 1)
            (= (cadr lst) 1)
        )
        (
            and
            (= (car lst) 6)
            (= (cadr lst) 6)
        )
    )
)

(defun gameplay2 ()
    (princ "Second player throw: ")
    (setq thr2 (bones_throw))
    (princ thr2)
    (terpri)
    (
        cond
        (
            (win_check thr2)
            (princ "Second player wins!")
        )
        (
            (pass_check thr2)
            (gameplay2)
        )
        (
            T
            (
                cond
                (
                    (
                        >
                        (+ (car thr1) (cadr thr1))
                        (+ (car thr2) (cadr thr2))
                    )
```

```
                    (princ "First player wins!")
                )
                (
                    (
                        <
                        (+ (car thr1) (cadr thr1))
                        (+ (car thr2) (cadr thr2))
                    )
                    (princ "Second player wins!")
                )
                (
                    T
                    (princ "Draw in the game!")
                )
            )
        )
    )
)
(defun gameplay1 ()
    (princ "First player throw: ")
    (setq thr1 (bones_throw))
    (princ thr1)
    (terpri)
    (
        cond
        (
            (win_check thr1)
            (princ "First player wins!")
        )
        (
            (pass_check thr1)
            (gameplay1)
        )
        (
            T
            (gameplay2)
        )
    )
    (terpri)
)
(gameplay1)
```

Задание 7

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

```
(defun palindrom_check (lst)
    (defun st_check (ls revls n)
        (
            cond
            (
                (> n 1)
                (
                    and
                    (= (car ls) (car revls))
                    (st_check (cdr ls) (cdr revls) (- n 2))
                )
            )
            (
                T
                T
            )
        )
    )
    (st_check lst (reverse lst) (length lst))
)
```

Задание 8

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

```
(defun countries_capitals (lst name)
    (
        cond
        (
            (
                equal
                (caar lst)
                name
            )
            (cdar lst)
        )
        (
            (
                equal
                (cdar lst)
                name
            )
            (caar lst)
        )
        (
            (cdr lst)
            (countries_capitals (cdr lst) name)
        )
    )
)
```

Задание 9

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

```
(defun mult_el_a (n lst)
    (
        cond
        (
            (
                and
```

```
7                   (
8                       and
9                       (numberp (car lst))
10                      (
11                          and
12                          (numberp (cadr lst))
13                          (numberp (caddr lst))
14                      )
15                  )
16                  (numberp n)
17              )
18              (* (car lst) n)
19          )
20          (
21              T
22              Nil
23          )
24      )
25  )
26
27  (defun mult_el_b (n lst)
28      (
29          cond
30          (
31              (
32                  and
33                  (numberp (car lst))
34                  (numberp n)
35              )
36              (* (car lst) n)
37          )
38          (
39              T
40              Nil
41          )
42      )
43  )
```