



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 курсу «Функциональное и логическое программирование»

на тему: «Использование управляющих структур, работа со списками»

Студент ИУ7-61Б  
(Группа)

\_\_\_\_\_  
(Подпись, дата)

Савинова М. Г.  
(Фамилия И. О.)

Преподаватель

\_\_\_\_\_  
(Подпись, дата)

Толпинская Н. Б.  
(Фамилия И. О.)

Преподаватель

\_\_\_\_\_  
(Подпись, дата)

Строганов Ю.В.  
(Фамилия И. О.)

2024 г.

# СОДЕРЖАНИЕ

<b>1</b>	<b>Практические задания</b>	<b>3</b>
1.1	Задание 1 . . . . .	3
1.2	Задание 2 . . . . .	3
1.3	Задание 3 . . . . .	4
1.4	Задание 4 . . . . .	4
1.5	Задание 5 . . . . .	5
1.6	Задание 6 . . . . .	5
1.7	Задание 7 . . . . .	7
1.8	Задание 8 . . . . .	7
1.9	Задание 9 . . . . .	8

# 1 Практические задания

## 1.1 Задание 1

Чем принципиально отличаются функции `cons`, `list`, `append`?

**Ответ:**

- `cons` — базовая функция, которая объединяет значения двух своих аргументов в точечную пару.
- `list` — принимает произвольное число аргументов и возвращает список, состоящий из значений аргументов.
- `append` — объединяет списки в один, при этом оригинальные списки остаются нетронутыми, и возвращается новый список, содержащий комбинированные элементы.

Каковы результаты вычисления следующих выражений?

```
1 (setf lst1 '(a b c))
2 (setf lst2 '(d e))
3
4 (print (cons lst1 lst2)) ;; ((A B C) D E)
5 (print (list lst1 lst2)) ;; ((A B C) (D E))
6 (print (append lst1 lst2)) ;; (A B C D E)
```

## 1.2 Задание 2

Каковы результаты вычисления следующих выражений, и почему?

```
1 (print (reverse '(a b c))) ;; (C B A)
2 (print (reverse '(a b (c (d)))))) ;; ((C (D)) B A)
3 (print (reverse '(a))) ;; (A)
4 (print (reverse ())) ;; NIL
5 (print (reverse '((a b c)))) ;; ((A B C))
6
7 (print (last '(a b c))) ;; (C)
8 (print (last '(a))) ;; (A)
9 (print (last '(a b (c)))) ;; ((C))
10 (print (last ())) ;; NIL
```

## Ответ:

- **reverse** работает не разрушающим образом и не изменяет исходный список, а возвращает новый список с элементами в обратном порядке. Работает только с элементами верхнего уровня;
- **last** используется для извлечения последнего элемента из списка. Она принимает один аргумент — список и возвращает последний элемент этого списка. Работает только с элементами верхнего уровня;

### 1.3 Задание 3

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

```
1 (defun last_1 (lst)
2   (last lst)
3 )
4
5 (defun last_2 (lst)
6   (cond ((null lst) Nil)
7         (t (cons (car (reverse lst)) Nil)))
8   )
9 )
10
11 (defun last_3 (lst)
12   (cond ((cdr lst) (last_3 (cdr lst)))
13         ((cond ((null lst) Nil)
14               (t (cons (car lst) Nil))))
15   )
16 )
17 )
```

### 1.4 Задание 4

Написать два варианта функции, которая возвращает свой список аргумента без последнего элемента.

```
1 (defun no_last_1 (lst)
2   (reverse (cdr (reverse lst)))
3 )
4
```

```

5 | (defun no_last_2 (lst)
6 |   (cond ((null (cdr lst)) Nil)
7 |         (t (cons (car lst) (no_last_2 (cdr lst)))))
8 | )

```

## 1.5 Задание 5

Написать функцию `swap-first-last`, которая переставляет в списке-аргументе первый и последний элемент.

```

1 |
2 | (defun get_middle (lst x)
3 |   (cond ((null lst) lst)
4 |         ((null (cdr lst)) (cons x Nil))
5 |         (t (cons (car lst) (get_middle (cdr lst) x))))
6 |   )
7 | )
8 |
9 | (defun swap (lst)
10 |   (cons (car (last lst)) (get_middle (cdr lst) (car lst))))
11 | )

```

## 1.6 Задание 6

Написать простой вариант игры в кости, в котором бросается 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. Результаты игры и значения выпавших костей выводить на экран с помощью `print`.

```

1 | (defun get_sum (dice)
2 |   (+ (car dice) (cdr dice)))
3 | )
4 |
5 | (defun roll_dice ()
6 |   (setf *random-state* (make-random-state t))
7 |   (cons (+ (random 5) 1)
8 |         (+ (random 5) 1)))
9 | )

```

```

10 )
11
12 (defun is_abs_win (dice)
13     (let (
14         (sum (get_sum dice))
15         )
16         (or (= sum 7) (= sum 11))
17     )
18 )
19
20 (defun is_reroll (dice)
21     (let (
22         (fir (car dice))
23         (sec (cdr dice))
24         )
25         (or (and (= fir 6) (= sec 6))
26             (and (= fir 1) (= sec 1)))
27     )
28 )
29
30 (defun get_dice_res (id)
31     (let (
32         (dices (roll_dice))
33         )
34         (print 'Player)
35         (princ id)
36         (print 'dices)
37         (princ dices)
38         (cond ((is_reroll dices)
39                 (print 'Rerolling)
40                 (get_dice_res id))
41               (t dices))
42     )
43 )
44
45 (defun game ()
46     (let (
47         (dice1 (get_dice_res 1))
48         (dice2 (get_dice_res 2))
49         )
50         (cond (

```

```

51         (is_abs_win dice1)
52         (print "Player1 won absolutely")
53     )
54     (
55         (is_abs_win dice2)
56         (print "Player2 won absolutely")
57     )
58     (
59         (> (get_sum dice1) (get_sum dice2))
60         (print "Player1 won")
61     )
62     (
63         (< (get_sum dice1) (get_sum dice2))
64         (print "Player2 won")
65     )
66     (
67         (= (get_sum dice1) (get_sum dice2))
68         (print "Draw")
69     )
70 )
71 )
72 )
73
74 (game)

```

## 1.7 Задание 7

Написать функцию, которая по своему списку-аргументу `lst` определяет является ли он палиндромом.

```

1 (defun pal (lst)
2   (equal lst (reverse lst)))
3 )

```

## 1.8 Задание 8

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

```

1 (defun both (tbl name)
2   (cond ((null tbl) Nil)

```

```

3         ((eq1 name (cdar tbl)) (caar tbl))
4         ((eq1 name (caar tbl)) (cdar tbl))
5         (t (f (cdr tbl) name))
6     )
7 )
8
9 (defun con_by_cap (tbl name)
10     (cond ((null tbl) Nil)
11           ((eq1 name (caar tbl)) (cdar tbl))
12           (t (f1 (cdr tbl) name))
13     )
14 )
15
16 (defun cap_by_con (tbl name)
17     (cond ((null tbl) Nil)
18           ((eq1 name (cdar tbl)) (caar tbl))
19           (t (f2 (cdr tbl) name))
20     )
21 )

```

## 1.9 Задание 9

Написать функцию, которая умножает на заданное число-аргумент первый числовой элемент списка их заданного 3-х элементного списка-аргумента, когда

- 1) все элементы списка — числа;
- 2) элементы списка — любые объекты.

```

1 (defun mul (lst x)
2     (cond ((null lst) Nil)
3           ((numberp (car lst)) (cons (* (car lst) x) (cdr lst)))
4           (t (cons (car lst) (mul_any (cdr lst) x)))
5     )
6 )

```