



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»

Отчёт по лабораторной работе №4 по курсу «Функциональное и логическое программирование»

Тема Использование управляющих структур, работа со списками

Студент Прянишников А.Н.

Группа ИУ7-65Б

Оценка (баллы) _____

Преподаватели Строганов Ю. В., Толпинская Н. Б.

Практическое задание

Задание 1

Каковы результаты вычисления следующих выражений?

```
1 (setf lst1 '(a b))
2 (setf lst2 '(c d))
3
4 (cons lst1 lst2) ; (a b) . (c d) -> ((a b) c d)
5 (list lst1 lst2) ; ((a b) (c d))
6 (append lst1 lst2) ; (a b c d)
```

Задание 2

Каковы результаты вычисления следующих выражений, и почему?

```
1 (reverse ()) ; NIL
2 (last ()) ; NIL
3 (reverse '(a)) ; (A)
4 (last '(a)) ; (A)
5 (reverse '((a b c))) ; ((A B C))
```

Задание 3

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

```
1 (defun last1 (a) (first (last a)))
2 (defun last2 (a) (first (reverse a)))
```

Задание 4

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

```
1 (defun delete_last (a)
2   (reverse (cdr (reverse a))))
3
4 (defun delete_last_2 (a)
5   (butlast a))
6
7 (defun delete_last_3(a)
8   (cond
9     ((null (cdr a)) nil)
10    (t (cons (car a) (reduce_list_3 (cdr a)))
11    ))
12 )
```

Задание 5

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 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.

```
1 (defun random_score ()
2   (list (+ (random 5) 1) (+ (random 5) 1)))
3
4 (defun check_sum_to_replay(result)
5   (if (or (equal result '(6 6)) (equal result '(1 1))) T NIL))
6
7 (defun print_result_to_replay (result)
8   (print "Score: ")
9   (prin1 result)
10  (print "The dice will be rerolled...")
11  (print "-----"))
12
13 (defun make_player_score ()
14  (let* ((result (random_score)))
```

```

15 (if (check_sum_to_replay result)
16     (and (print_result_to_replay result)
17         (make_player_score) result)))
18
19 (defun sum (result)
20 (+ (first result) (second result)))
21
22 (defun check_sum_to_win (result)
23 (if (or (equal (sum result) 7)
24 (equal (sum result) 11)) T NIL))
25
26 (defun print_player (result)
27 (print "Score: ")
28 (PRIN1 result))
29
30 (defun play()
31 (print "---First player---")
32 (let* ((first_val (make_player_score)))
33     (if (check_sum_to_win first_val) (and (print_player first_val) "First player won!")
34         (and (print_player first_val)
35             (print "-----")
36             (print "---Second player---")
37             (let* ((second_val (make_player_score)))
38                 (if (check_sum_to_win second_val) (and (print_player second_val) "Second
39                     player won!")
40                     (and (print_player second_val)
41                         (if (>= (sum first_val) (sum second_val))
42                             "First player won!" "Second player won!")
43                         ))))))))

```

Контрольные вопросы

1. Синтаксическая форма и хранение программы в памяти

В LISP формы представления программы и обрабатываемых ею данных одинаковы и представляются в виде **S-выражений**. Поэтому программы могут обрабатывать и преобразовывать другие программы и даже самих себя. В процессе трансляции можно введенное и сформированное в результате вычислений выражение данных проинтерпретировать в качестве програм-

мы и непосредственно выполнить. Так как программа представляет собой S-выражение, в памяти она представлена либо как атом (5 указателей; форма представления атома в памяти), либо списковой ячейкой (бинарный узел; 2 указателя).

2. Трактовка элементов списка

Первый аргумент списка, который поступает на вход интерпретатору, трактуется как имя функции, остальные – как аргументы этой функции.

3. Порядок реализации программы

Программа в языке LISP представляется S-выражением, которое передается интерпретатору – функции `eval`, которая выводит последний, полученный после обработки S-выражения, результат. Работа функции `eval` представлена на картинке ниже.

4. Способы определения функций

С помощью макро определения `defun` или с использованием Лямбда-нотации (функция без имени).