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 управляющих структур, работа со списками

Студент Криков А.В.

Группа ИУ7-63Б

Оценка (баллы) _____

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

Москва — 2022 г.

Практические задания

Задание 1

Каковы результаты вычисления следующих выражений?

```
(setf lst1 '(a b))
(setf lst2 '(c d))

(cons lst1 lst2)    ; (a b) . (c d) ->  ((a b) c d)
(list lst1 lst2)    ; ((a b) (c d))
(append lst1 lst2) ; (a b c d)
```

Задание 2

Каковы результаты вычисления следующих выражений, и почему?

```
(reverse ())          ; NIL
(last ())             ; NIL
(reverse '(a))        ; (A)
(last '(a))           ; (A)
(reverse '((a b c))) ; ((A B C))
```

Задание 3

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

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

Задание 4

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

```
(defun delete_last (a)
(reverse (cdr (reverse a))))

(defun delete_last_2 (a)
(butlast a))

(defun delete_last_3(a)
  (cond
    ((null (cdr a)) nil)
    (t (cons (car a) (reduce_list_3 (cdr a)))
    ))
)
```

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.

```
(defun random_score ()
(list (+ (random 5) 1) (+ (random 5) 1)))

(defun check_sum_to_replay(result)
(if (or (equal result '(6 6)) (equal result '(1 1))) T NIL))

(defun print_result_to_replay (result)
(print "Score: ")
(prin1 result)
(print "The dice will be rerolled...")
(print "———————"))

(defun make_player_score ()
(let* ((result (random_score)))
    (if (check_sum_to_replay result)
        (and (print_result_to_replay result)
            (make_player_score)) result)))

(defun sum (result)
(+ (first result) (second result)))

(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 player won!")
39                  (and (print_player second_val)
40                      (if (>= (sum first_val) (sum second_val))
41                          "First player won!" "Second player won!")
42                      )))))))
```

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 программы 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).