|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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»\_\_\_\_\_\_\_\_\_\_\_

**Лабораторная работа № 7**

|  |  |
| --- | --- |
| **Дисциплина** Функциональное и логическое программирование  **Тема** \_Функционалы\_  **Студент** \_Ильясов И. М.\_  **Группа** \_ИУ7-63Б\_  **Оценка (баллы)** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  **Преподаватель** \_Толпинская Н. Б.\_ |  |

Москва, 2020 г.

**Задание 1.** Чем принципиально отличаются функции cons, list, append?

Пусть

(setf lst1 '( a b))

(setf lst2 '( d)).

Каковы результаты вычисления следующих выражений?

(cons lstl lst2) => ((A B) D)

(list lst1 lst2) => ((A B) (D))

(append lst1 lst2) => (A B D)

**Задание 2.** Каковы результаты вычисления следующих выражений?

(reverse ()) => Nil

(last ()) => Nil

(reverse '(a)) => (A)

(last '(a)) => (A)

(reverse '((a b c))) => (A B C)

(last '((a b c))) => (A B C)

**Задание 3.** Написать, по крайней мере, два варианта функции, которая возвращает последний элемент своего списка-аргумента.

(defun get-last1 (lst)

(car (reverse lst)))

(defun get-last2 (x)

(if (equal (cdr x) nil)

(car x)

(get-last2 (cdr x))

)

)

**Задание 4.** Написать, по крайней мере, два варианта функции, которая возвращает свой список-аргумент без последнего элемента.

(defun cut2 (x)

(cons (car x)

(if (> (length (cdr x)) 1)

(cut2 (cdr x))

nil

)

)

)

**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 roll-dice () (list (+ (random 6) 1) (+ (random 6) 1)))

(defun play-sum (res)

(+ (first res) (second res))

)

(defun is-win (res)

(or (equal 7 (play-sum res))

(equal 11 (play-sum res))

)

)

(defun is-lucky (res)

(or (equal res '(1 1))

(equal res '(6 6))

)

)

(defun play ()

(setq player1 (roll-dice))

(format T " First player ~A" player1)

(if (equal T (is-win player1))

(print " First player win!")

(if (equal T (is-lucky player1))

(play)

(last

(list

(setq player2 (roll-dice))

(format T " Second player ~A" player2)

(if (= (play-sum player1) (play-sum player2))

(print "Tie!")

(if (> (play-sum player1) (play-sum player2))

(print "First player win!")

(print "Second player win!")

)

)

)

)

)

)

)