Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский государственный технический университет имени Н.Э.Баумана»

(МГТУ им. Н.Э.Баумана)



*Факультет «Информатика и системы управления»*

*Кафедра «Программное обеспечение ЭВМ и информационные технологии»*

**Дисциплина: Функциональное и логическое программирование**

**Лабораторная работа №5**

Выполнил:

Левушкин Илья

ИУ7-62Б

Москва, 2020

**1. Написать функцию, которая принимает целое число и возвращает первое**

**четное число, не меньшее аргумента.**

(defun near\_even (ar) (

cond ((oddp ar) (+ ar 1))

(T ar)

))

Примеры:

1. (near\_even 3) = 4
2. (near\_even 2) = 2
3. (near\_even 33) = 34

**2. Написать функцию, которая принимает число и возвращает число**

**того же знака, но с модулем на 1 больше модуля аргумента.**

(defun abs\_more (ar) (

cond ((< ar 0) (- ar 1))

(T (+ ar 1))

))

Примеры:

1. (abs\_more 6) = 7
2. (abs\_more -6) = -7
3. (abs\_more 0) = 1

**3. Написать функцию, которая принимает два числа и возвращает**

**список из этих чисел, расположенный по возрастанию.**

(defun list\_increase (ar1 ar2) (

cond ((< ar1 ar2) (list ar1 ar2))

(T (list ar2 ar1))

))

Примеры:

1. (list\_increase 9 0) = (0 9)
2. (list\_increase 9 9) = (9 9)
3. (list\_increase 0 7) = (0 7)

(defun list\_increase (ar1 ar2) (

if (> ar1 ar2) (cons ar2 (cons ar1 Nil))

(cons ar1 (cons ar2 Nil))

))

Примеры:

1. (list\_increase 9 0) = (0 9)
2. (list\_increase 9 9) = (9 9)
3. (list\_increase 0 7) = (0 7)

**4. Написать функцию, которая принимает три числа и возвращает Т только**

**тогда, когда первое число расположено между вторым и третьим.**

(defun middle (ar1 ar2 ar3) (

cond ((and (< ar1 ar3) (> ar1 ar2)) T)

(T Nil)

))

Примеры:

1. (middle 3 1 2) = NIL
2. (middle 1 2 3) = NIL
3. (middle 2 1 3)= T

**5. Каков результат вычисления следующих выражений?**

|  |  |
| --- | --- |
| **(and 'fee 'fie 'foe) = foe** | **(or 'fee 'fie 'foe) = fee** |
| **(or nil 'fie 'foe) = fie** | **(and nil 'fie 'foe) = Nil** |
| **(and (equal 'abc 'abc) 'yes) = yes** | **(or (equal 'abc 'abc) 'yes) = T** |

**6. Написать предикат, который принимает два числа-аргумента и возвращает**

**Т, если первое число не меньше второго.**

(defun not\_less (ar1 ar2) (

cond ((< ar1 ar2) Nil)

(T T)

))

Примеры:

1. (not\_less 0 1) = Nil
2. (not\_less 1 1) = T
3. (not\_less 2 1) = T

**7. Какой из следующих двух вариантов предиката ошибочен и почему?**

**(defun pred1 (x) (defun pred2 (x)**

**(and (numberp x) (plusp x))) (and (plusp x)(numberp x)))**

Функция pred2 написана неверно, так как если на вход ей будет подано не число, а строка, то произойдет ошибка в функции plusp. А если передать строку в предикат pred1, то сначала произойдет проверка на число.

**8. Решить задачу 4, используя для ее решения конструкции**

**IF, COND, AND/OR.**

(defun middle\_cond (ar1 ar2 ar3) (

cond (

(< ar1 ar3) (cond (

(> ar1 ar2) T

)

(T Nil)

)

)

(T Nil)

))

(defun middle\_and (ar1 ar2 ar3) (

and (< ar1 ar3) (> ar1 ar2)

))

(defun middle\_if (ar1 ar2 ar3) (

if (< ar1 ar3) (if (> ar1 ar2) T Nil) Nil

))

**Ответы на вопросы:**

1) Атом. Определение, представление в памяти

Атомы - простейшие объекты Лиспа, из которых строятся остальные структуры.

Атомы:

1. символы - набор литер (букв и цифр)
2. специальные символы - T, Nil, используются для обозначения логических констант
3. самоопределимые атомы - натуральные, дробные числа, строки - последовательность символов, заключенных в двойные ковычки.

Атомы в памяти представлены с помощью 5 указателей.

2) Самовычислимый атом

Самоопределимые атомы — натуральные числа, дробные числа, вещественные числа, строки — последовательность символов, заключенных в двойные апострофы.

3) eval и quote

Функция EVAL возвращает результат выражения <выражение>, где <выражение> - любое выражение языка LISP.

QUOTE возвращает выражение не выполняя его.

Quote и eval действуют во взаимно противоположенных направлениях и аннулируют эфефект друг друга.

4) Глобальные и локальные символьные атомы

Глобальные символьные атомы — значение устанавливается с помощью setf. Область видимости весь код следующий после определения.

(setf a 9)

a

9

Локальные значение — значение устанавливается с помощью let(let\*)

Область видимости является тело функции, в которой определена переменная.

(let ((x 1) (y 2))

(+ x y))

3