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 №5

Студент: Луговой Д.М.
Группа: ИУ7-61Б
Преподаватель: Толпинская Н.Б.

Москва, 2020 г.

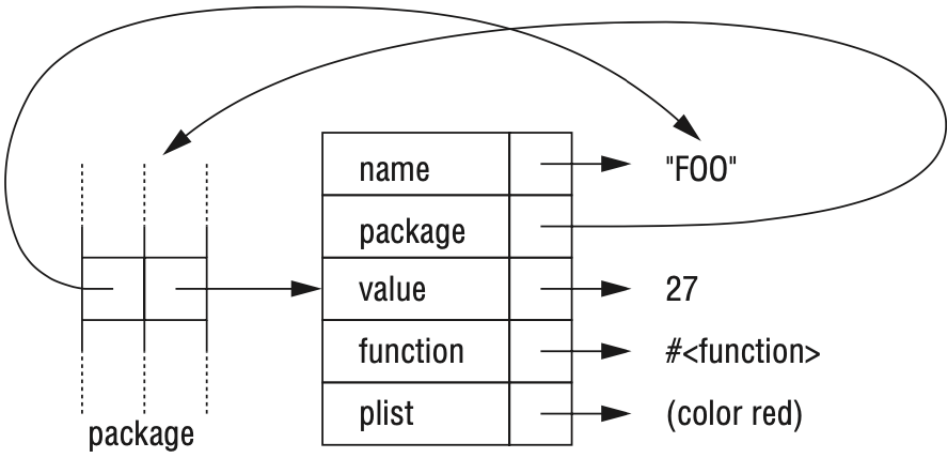
Цель работы: приобрести навыки работы в Common Lisp.

Задачи работы: изучить работу интерпретатора Lisp, алгоритм работы функции eval, структуру и порядок обработки программы в Lisp.

1. **Представление атома в памяти** Атомы - это элементарные элементы Lisp, из которых строятся остальные структуры. Атомы в Lisp:

   - Символы - последовательность из букв и цифр, начинающаяся с буквы, включая и другие литеры, не занятые в синтаксисе.
   - Специальные символы - логические константы T и Nil.
   - Самовычислимые атомы - натуральные, дробные и вещественные числа и строки.

   В памяти атом представлен как структура из 5 указателей:



2. **Самовычисляемые атомы**

   В Lisp натуральные, дробные и вещественные числа и строки, а также константы Nil и T являются самовычисляемыми. Это означает, что когда данный объект вычисляется, он (или возможно копия в случае с числами и строковыми символами) возвращается в качестве значения.

   Например, при вычислении атомов 10 или "Hello, world!"эти атомы будут вычислены в 10 и "Hello, world!"соответсвенно.

3. **Локальное и глобальное определение значения атома**

   Локальные значение атома можно определить с помощью функций let и let*. Областью видимости является тело функции, в которой определена переменная.

Синтаксис:

(let ((var1 value1) (var2 value2) ... (varN valueN)) body)

Сначала вычисляются значения value1, value2, ... , valueN, а затем происходит их связывание с var1, var2, ... , varN.

(let* ((var1 value1) (var2 value2) ... (varN valueN)) body)

Отличие от let состоит в том, что связывание каждого значения value с символом var происходит сразу после вычисления значения.

Глобальные значение атома устанавливается с помощью функции setf. Областью видимости является весь код, следующий после определения.

Синтаксис:

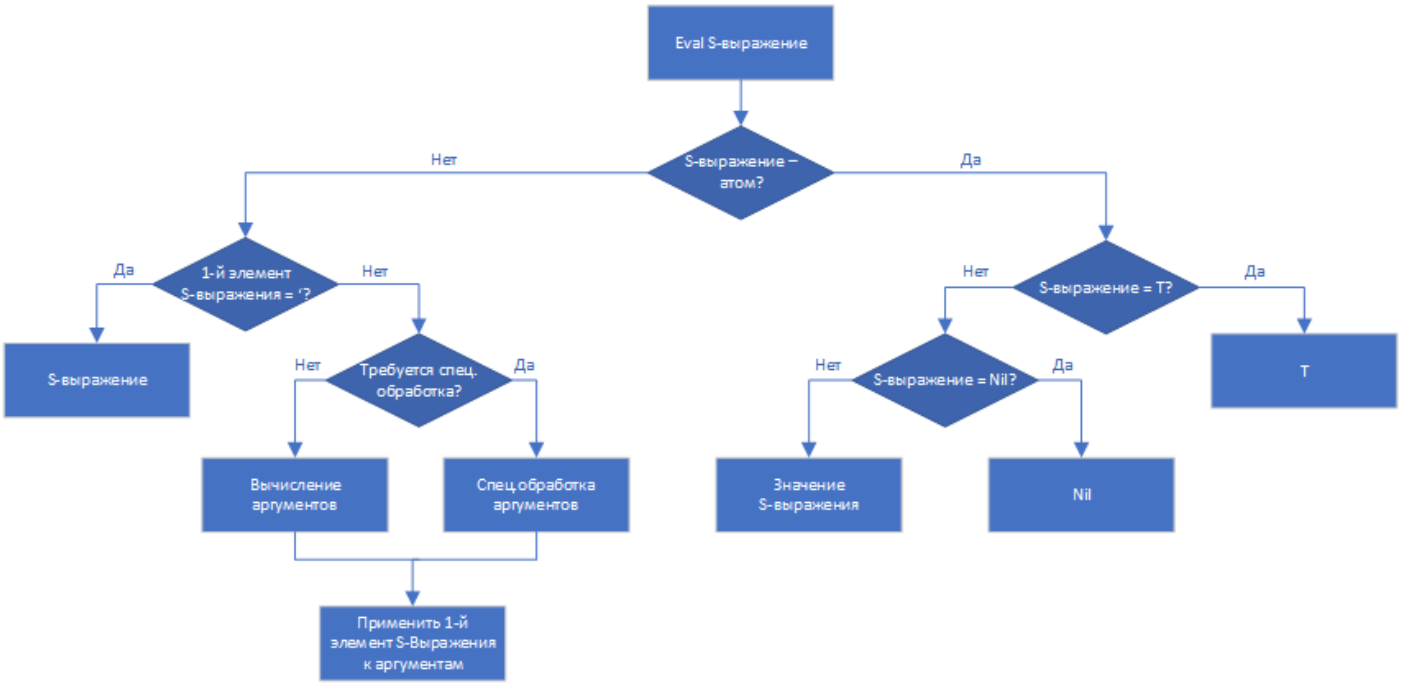
(setf var value)

4. **Функции EVAL и QUOTE**

Функция eval, вычисляет заданное выражение и возвращает его значение:

(eval '(+ 1 2 3)) -> 6

Схема выполнения Eval:



И программа, и данные в Lisp представляются в списочной форме. Для явного выделения данных используется функция QUOTE и ' - ее сокращенное обозначение. Эта функция защищает выражение от вычисления.

Если eval была применена явно, блокировка вычисления quote не сработает, так как eval обеспечивает дополнительный вызов интерпретатора.

Задание №1

Написать функцию, которая принимает целое число и возвращает первое четное число, не меньшее аргумента.

(defun get_even (x) (cond ((oddp x)(+ x 1)) (x)))

Примеры:

- (get_even 4) -> 4
- (get_even 7) -> 8

Задание №2

Написать функцию, которая принимает число и возвращает число того же знака, но с модулем на 1 больше модуля аргумента.

(defun inc_abs (x) (cond ((plusp x)(+ x 1))((- x 1))))

Примеры:

- (inc_abs 3) -> 4
- (inc_abs -5) -> -6

Задание №3

Написать функцию, которая принимает два числа и возвращает список из этих чисел, расположенный по возрастанию.

(defun get_asc (x y) (cond ((> x y)(list y x))((list x y))))

Примеры:

- (get_asc 3 5) -> (3 5)
- (get_asc 5 -1) -> (-1 5)

Задание №4

Написать функцию, которая принимает три числа и возвращает T только тогда, когда первое число расположенно между вторым и третьим.

(defun between (a b c) (or (and (> a b) (< a c)) (and (< a b) (> a c))))

Примеры:

- (between 8 7 10) -> T
- (between 2 3 0) -> T
- (between 1 5 8) -> NIL

Задание №5

Каков результат вычисления следующих выражений?

1. (and 'fee 'fie 'foe)
   Результат: FOE
2. (or 'fee 'fie 'foe)
   Результат: FEE
3. (and (equal 'abc 'abc) 'yes)
   Результат: YES
4. (or nil 'fie 'foe)
   Результат: FIE
5. (and nil 'fie 'foe)
   Результат: NIL
6. (or (equal 'abc 'abc) 'yes)
   Результат: T

Задание №6

Написать предикат, который принимает два числа-аргумента и возвращает T, если первое число не меньше второго.

(defun is_greater (x y) (>= x y))

Примеры:

- (is_greater 5 2) -> T
- (is_greater 7 20) -> NIL
- (is_greater 6 6) -> T

Задание №7

Какой из следующих двух вариантов предикатов ошибочен и почему?

1. (defun pred1 (x) (and (numberp x) (plusp x)))
2. (defun pred2 (x) (and (plusp x) (numberp x)))

Ошибочен второй предикат, так как в нем сначала вычисляется предикат plusp, который может вернуть ошибку в случае, если x не является числом, а первый предикат в этом случает вычислит предикат numberp, вернет NIL и предикат plusp вычислен не будет.

Задание №8

Решить задачу 4, используя для ее решения конструкции:

- COND
  (defun between (a b c) (cond ((cond ((> a b) (< a c)) ((< a b) (> a c))))))
- IF
  (defun between (a b c) (if (> a b) (< a c) (if (< a b) (> a c))))
- AND/OR
  (defun between (a b c) (or (and (> a b) (< a c)) (and (< a b) (> a c))))