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 №2

Название «Определение функций пользователя»

Дисциплина «Функциональное и логическое программирование»

Студент ИУ7-65Б _____ (подпись, дата) Бугаенко А.П. (Фамилия И.О.)

Преподователь _____ (подпись, дата) Толпинская Н.Б. (Фамилия И.О.)

Москва, 2022

1 Цели и задачи работы

Цель работы — приобрести навыки создания и использования функций пользователя в Lisp. Задачи работы — изучить работу интерпретатора Lisp, алгоритм работы функции eval, структуру и порядок обработки программы в Lisp.

2 Теоретические вопросы

2.1 Базис Lisp

Базис Lisp составляют атомы, структуры, базовые функции и встроенные и спецальные функционалы.

2.2 Классификация функций

Функция - однозначное отображение множества значений аргументов в значение функции.
Функциональный язык - тот, который базируется на понятии функции.
Функциональность системы - предоставляемые пользователю возможности.

Классификация функций по аргументам и поведению.

— чистые функции (математические функции) - имеют фиксированное число аргументов, для определённого набора аргументов один фиксированный результат

— формы (специальные функции) - функции, принимающие на вход произвольное количество аргументов, или по-разному обрабатывающее аргументы.

— псевдо-функции - функции, обладающие побочным эффектом. Побочный эффект - событие, изменяющее сознание системы. Пример - setf, связывающее атом и значение и format, выводящее значение на экран.

— функционалы - принимают функции в качестве параметров либо в качестве возвращаемого значения выступает функция.

Классификация функций по именованию.

— именованные - есть имя, определяется через defun. Специальные символы (T, Nil) и самоопределимые атомы (натуральные, вещественные числа, строки) не могут выступать в роли функции.

— неименованные - нет имени, через lambda.

2.3 Способы создания функций

Создание именованной функции - синтаксис:
(defun имя список_аргументов лямбда-выражение)

Создание неименованной функции - синтаксис:
(lambda список_аргументов лямбда-выражение)
(lambda $(x_1, ..., x_k)$ форма)

2.4 Функции car и cdr

car и cdr - базовые функции доступа к данным, хранящихся в формате списка.
car - принимает точечную пару или список в качестве аргумента и возвращает первый элемент (голову, значение по car-указателю) или nil, если на вход был подан пустой список.

cdr - принимает точечную пару или список и возвращает хвост (значение по cdr-указателю). Если список, поданный на вход непустой, то возвращается список из всех элементов, кроме первого. Если пустой, возвращается Nil.

2.5 Назначение и отличия в работе cons и list

cons - создаёт списковую ячейку и ставит указатели на два аргумента, таким образом объединяя свои аргументы в точечную пару. list - создаёт список из значений поданных на вход аргументов, причём количество аргументов может быть произвольным.

3 Практические задания

3.1 Задание 1

Составить диаграмму вычисления следующих выражений:

3.2 Задание 2

Написать функцию, вычисляющую гипотенузу прямоугольного треугольника по заданным катетам и составить диаграмму её вычисления.

```
(defun hypot (a b) (sqrt (+ (* a a) (* b b))))
```

3.3 Задание 3

Написать функцию, вычисляющую объем параллелепипеда по 3-м его сторонам, и составить диаграмму ее вычисления.
(defun volume (a b c) (* a b c))

3.4 Задание 4

Каковы результаты вычисления следующих выражений?(объяснить возможную ошибку и варианты ее устранения)

(list 'a c) → ошибка, атом c не связан ни с каким значением
(cons 'a (b c)) → ошибка, список не передаётся через quote, и атомы b, c не связаны ни с каким значением
(cons 'a '(b c)) → (a b c)
(caddy (1 2 3 4 5) → ошибка, некорректный вызов функции, атом caddy не связан ни с каким лямбда-выражением
(cons 'a 'b 'c) → ошибка, cons принимает на вход только два аргумента
(list 'a (b c)) → ошибка, список не передаётся через quote, и атомы b, c не связаны ни с каким значением
(list a '(b c)) → ошибка, атом a не связан ни с каким значением
(list (+ 1 '(length '(1 2 3)))) → ошибка, выражение '(length '(1 2 3)) не является числом

3.5 Задание 5

Написать функцию longer_then от двух списков-аргументов, которая возвращает T, если первый аргумент имеет большую длину.

(defun longer_then (a b) (> (length a) (length b)))

3.6 Задание 6

Каковы результаты вычисления следующих выражений?

(cons 3 (list 5 6)) → (3 5 6)
(cons 3 '(list 5 6)) → (3 list 5 6)
(list 3 'from 9 'lives (- 9 3)) → (list 3 'from 9 'lives (- 9 3))
(+ (length for 2 too)) (car '(21 22 23))) → ошибка, атомы for и too не связаны ни с какими значениями
(cdr '(cons is short for ans)) → (cdr '(cons is short for ans))
(car (list one two)) → ошибка, атомы one и too не связаны ни с какими значениями
(car (list 'one 'two)) → one

3.7 Задание 7

Дана функция (defun mystery (x) (list (second x) (first x))). Какие результаты вычисления следующих выражений?

(mystery (one two)) → ошибка, атомы one и two не связаны ни с какими значениями
(mystery one 'two)) → ошибка, функция принимает на вход только один аргумент
(mystery (last one two)) → ошика, на вход функции last должен подаваться список
(mystery free) → ошибка, атом free не связан ни с каким значением

3.8 Задание 8

Написать функцию, которая переводит температуру в системе Фаренгейта температуру по Цельсию (defum f-to-c (temp)...). Формулы: c = 5/9*(f-320); f= 9/5*c+32.0. Как бы назывался роман Р.Брэдбери "+451 по Фаренгейту"в системе по Цельсию?

(defun f-to-c (temp) (* (/ 5 9) (- temp 32.0)))
"+451 по Фаренгейту"→ "+232.77779 градус по Цельсию"

3.9 Задание 9

Что получится при вычисления каждого из выражений? (list 'cons t NIL) → (cons t Nil)
(eval (list 'cons t NIL)) → (t)
(eval (eval (list 'cons t NIL))) → ошибка, t - не функция
(apply #cons "(t NIL)) → ошибка, возможно имелось ввиду (apply #'cons '(t NIL)), в этом случае (t)
(eval NIL) → Nil
(list 'eval NIL) → (eval Nil)
(eval (list 'eval NIL)) → Nil