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 по дисциплине «Функциональное и логическое программирование»

Тема: Определение функций пользователя
Студент: Княжев А. В.
Группа: ИУ7-62Б
Оценка (баллы): _____
Преподаватели: Толпинская Н. Б., Строганов Ю. В.

Москва — 2023 г.

Оглавление

1. Теоретическая часть

1.1. Базис языка Lisp

Базис — это минимальный набор правил/конструкций языка, к которым могут быть сведены все остальные. Базис языка Lisp представлен атомами, структурами, базовыми функциями, базовыми функционалами. Некоторые базисные функции: `car`, `cdr`, `cons`, `quote`, `eq`, `eval`.

1.2. Классификация функций языка Lisp

Функции в Lisp бывают базисными, пользовательскими и функциями ядра. Также функции можно разделить на:

— чистые — не создающие побочных эффектов, принимающие фиксированное число аргументов, не получающие данные неявно, результат работы которых не зависит от внешних переменных;

— особые, или формы;

— функции более высоких порядков, или функционалы — функции, результатом и/или аргументом которых является функция.

1.3. Способы создания функций в языке Lisp

Функцию можно определить двумя способами: неименованную с помощью `lambda` и именованную с помощью `defun`.

```
(lambda (x_1 x_2 ... x_n) f)
```

— `f` — тело функции;

— `x_i` — формальные параметры.

```
(defun <имя> [lambda] (x_1 x_2 ... x_n) f)
```

— `f` — тело функции;

— `x_i` — формальные параметры.

Тогда имя будет ссылкой на описание функции.

1.4. Функции `car`, `cdr`, `eq`, `eql`, `equal`, `equalp`

Функции `car`, `cdr` являются базовыми функциями доступа к данным.

— `car` принимает точечную пару или список в качестве аргумента и возвращает указатель на первый элемент (если список пустой, то `Nil`).

— `cdr` принимает точечную пару или список в качестве аргумента и возвращает все элементы, кроме первого или `Nil` (указатель на хвост списка).

Функции сравнения — `eq`, `eql`, `equal`, `equalp`.

— `eq` возвращает истину тогда и только тогда, когда ее аргументы соответствуют одному и тому же объекту в памяти.

— `eql` возвращает истину, если его аргументы равны с точки зрения `eq`, или если это числа одинакового типа и с одинаковыми значениями, или если это одинаковые буквы.

— `equal` возвращает истину, если его аргументы равны с точки зрения `eql`, либо являются списковыми ячейками, чьи `car` и `cdr` эквивалентны с точки зрения `equal`, либо являются строками.

— `equalp` возвращает истину, если его аргументы равны с точки зрения `equal`, либо являются списковыми ячейками, чьи `car` и `cdr` эквивалентны с точки зрения `equalp`, либо являются списками одинаковой длины, элементы которых эквивалентны с точки зрения `equalp`.

1.5. Назначение и отличие в работе `cons` и `list`

Функции `list`, `cons` являются функциями создания списков (`cons` — базисная, `list` — нет).

— `cons` принимает два аргумента (первый необязательно список, второй список), создает списковую ячейку и устанавливает два указателя на аргументы. Если второй аргумент `cons` — атом, то формируется точечная пара.

— `list` принимает переменное число аргументов, создаёт списковые ячейки, количество которых соответствует количеству переданных параметров, и расставляет указатели. Возвращает список, элементы которого — переданные в функцию аргументы.

Отличия

— количество аргументов (у `cons` — фиксированное, у `list` — переменное);

— результат (у `cons` — списковая ячейка, у `list` — список);

— реализация доступа к памяти.

2. Практическая часть

2.1. Задание 1

Задание

Составить диаграмму вычисления следующих выражений:

1. `(equal 3 (abs - 3))`
2. `(equal (+ 1 2) 3)`
3. `(equal (* 4 7) 21)`
4. `(equal (* 2 3) (+ 7 2))`
5. `(equal (- 7 3) (* 3 2))`
6. `(equal (abs (- 2 4)) 3))`

Решение

Приложено на отдельном листе.

2.2. Задание 2

Задание

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

Решение

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

Диаграмма приложена на отдельном листе.

2.3. Задание 3

Задание

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

1. `(list 'a c)`
2. `(cons 'a (b c))`
3. `(cons 'a '(b c))`
4. `(caddr (1 2 3 4 5))`
5. `(cons 'a 'b 'c)`
6. `(list 'a (b c))`
7. `(list a '(b c))`
8. `(list (+ 1 '(length '(1 2 3))))`

Решение

1. `SYSTEM::READ-EVAL-PRINT: variable C has no value` — `C` воспринимается как переменная.
2. `EVAL: undefined function B` — `B` воспринимается как функция.
3. `(A B C)`.
4. `EVAL: 1 is not a function name; try using a symbol instead` — `1` воспринимается как функция.
5. `EVAL: too many arguments given to CONS: (CONS 'A 'B 'C)` — слишком много аргументов у функции.
6. `EVAL: undefined function B` — `B` воспринимается как функция.
7. `SYSTEM::READ-EVAL-PRINT: variable A has no value` — `A` воспринимается как переменная.
8. `+: (LENGTH '(1 2 3)) is not a number` — `(LENGTH '(1 2 3))` должно быть числом.

2.4. Задание 4

Задание

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

Решение

```
(defun longer_than (a b) (> (length a) (length b)))
```

2.5. Задание 5

Задание

Каковы результаты вычисления следующих выражений?

1. `(cons 3 (list 5 6))`
2. `(list 3 'from 9 'lives (- 9 3))`
3. `(+ (length for 2 too)) (car '(21 22 23)))`
4. `(cdr '(cons is short for ans))`
5. `(car (list one two))`
6. `(cons 3 '(list 5 6))`
7. `(car (list 'one 'two))`

Решение

1. `(3 5 6)`
2. `(3 FROM 9 LIVES 6)`
3. `SYSTEM::READ-EVAL-PRINT: variable FOR has no value`
4. `(IS SHORT FOR ANS)`

5. `SYSTEM::READ-EVAL-PRINT: variable ONE has no value`
6. `(3 LIST 5 6)`
7. `ONE`

2.6. Задание 6

Задание

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

1. `(mystery (one two))`
2. `(mystery (last one two))`
3. `(mystery free)`
4. `(mystery one 'two))`

Решение

1. `EVAL: undefined function ONE`
2. `SYSTEM::READ-EVAL-PRINT: variable ONE has no value`
3. `SYSTEM::READ-EVAL-PRINT: variable FREE has no value`
4. `SYSTEM::READ-EVAL-PRINT: variable ONE has no value`

2.7. Задание 7

Задание

Написать функцию, которая переводит температуру в системе Фаренгейта температуру по Цельсию.

Решение

```
(defun f_to_c (f) (* 5/9 (- f 320)))
```

2.8. Задание 8

Задание

Что получится при вычисления каждого из выражений?

1. `(list 'cons t NIL)`
2. `(eval (eval (list 'cons t NIL)))`
3. `(apply #cons "(t NIL))`
4. `(list 'eval NIL)`
5. `(eval (list 'cons t NIL))`
6. `(eval NIL)`
7. `(eval (list 'eval NIL))`

Решение

1. `(CONS T NIL)`
2. `EVAL: undefined function T`
3. `READ from #<INPUT CONCATENATED-STREAM #<INPUT STRING-INPUT-STREAM> #<IO TERMINAL-STREAM»: bad syntax for complex number: #CONS`
4. `(EVAL NIL)`
5. `(T)`
6. `NIL`
7. `NIL`