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 к лабораторной работе №3 по курсу: «Функциональное и логическое программирование»

Тема: **Определение функций пользователя**

Студент группы **ИУ7-62Б** **А.П. Сорокин**
(И.О. Фамилия)

Преподаватель **Н.Б. Толпинская**
(И.О. Фамилия)

2020 г.

СОДЕРЖАНИЕ

1 Цели и задачи

Цель работы: приобрести навыки создания и использования функций пользователя в Lisp.

Задачи работы: изучить способы создания и использования именованных и неименованных функций пользователя для обработки списков.

2 Теоретическая часть

Классификация функций:

- чистые (чисто математические): принимают строго определённое число аргументов и возвращают одно значение
- формы (произвольные): могут принимать разное число аргументов, в зависимости от чего по-разному себя ведёт
- функционалы: принимают функциональные описания.

Классификация базисных функций Lisp:

- функции-селекторы (функции доступа): car, cdr
- функции-конструкторы: cons, list
- функции-предикаты (функции проверки).

3 Практическая часть

3.1 Задание 1

Составить диаграмму вычисления для указанных выражений.

Задание 1.1. (equal 3 (abs - 3))

⟹ (equal 3 (abs - 3)):
  ⟹ 3
  ⟹ (abs - 3):
    ⟹ -3:
    ⟶ применение - к 3
      ⟵ возвращаемый результат: -3
  ⟶ применение abs к -3
   ⟵ возвращаемый результат: 3
⟶ применение equal к 3, 3
 ⟵ возвращаемый результат: T

Задание 1.2. (equal (+ 1 2) 3)

⟹ (equal (+ 1 2) 3):
  ⟹ (+ 1 2):
  ⟶ применение + к 1, 2
   ⟵ возвращаемый результат: 3
  ⟹ 3
⟶ применение equal к 3, 3
 ⟵ возвращаемый результат: T

Задание 1.3. (equal (* 4 7) 21)

⟹ (equal (* 4 7) 21):
  ⟹ (* 4 7):
  ⟶ применение * к 4, 7
   ⟵ возвращаемый результат: 28
  ⟹ 21
⟶ применение equal к 28, 21
 ⟵ возвращаемый результат: NIL

Задание 1.4. (equal (* 2 3) (+ 7 2))

⟹ (equal (* 2 3) (+ 7 2)):
  ⟹ (* 2 3):
  ⟶ применение * к 2, 3
  ⟵ возвращаемый результат: 6
  ⟹ (+ 7 2):
  ⟶ применение + к 7, 2
  ⟵ возвращаемый результат: 9
⟶ применение equal к 6, 9
⟵ возвращаемый результат: NIL

Задание 1.5. (equal (- 7 3) (* 3 2))

⟹ (equal (- 7 3) (* 3 2)):
  ⟹ (- 7 3):
  ⟶ применение - к 7, 3
  ⟵ возвращаемый результат: 4
  ⟹ (* 3 2):
  ⟶ применение * к 3, 2
  ⟵ возвращаемый результат: 6
⟶ применение equal к 4, 6
⟵ возвращаемый результат: NIL

Задание 1.6. (equal (- 7 3) (* 3 2))

⟹ (equal (abs (- 2 4)) 3):
  ⟹ (abs (- 2 4)):
    ⟹ (- 2 4):
    ⟶ применение - к 2, 4
    ⟵ возвращаемый результат: -2
  ⟶ применение abs к -2
  ⟵ возвращаемый результат: 2
  ⟹ 3
⟶ применение equal к 2, 3
⟵ возвращаемый результат: NIL

3.2 Задание 2

Функция, вычисляющая гипотенузу прямоугольного треугольника по заданным катетам:

Листинг 1 – Функция вычисления гипотенузы

```
1     (defun hypot (cath1 cath2) (sqrt (+ (expt cath1 2) (expt cath2 2))))
```

Диаграмма вычисления функции:

$\Longrightarrow$ (hypot 3 4)
  $\Longrightarrow$ (sqrt (+ (expt 3 2) (expt 4 2))):
    $\Longrightarrow$ (+ (expt 3 2) (expt 4 2)):
      $\Longrightarrow$ (expt 3 2):
      $\longrightarrow$ применение expt к 3, 2
       $\longleftarrow$ возвращаемый результат: 9
      $\Longrightarrow$ (expt 4 2):
      $\longrightarrow$ применение expt к 4, 2
       $\longleftarrow$ возвращаемый результат: 16
    $\longrightarrow$ применение + к 9, 16
     $\longleftarrow$ возвращаемый результат: 25
  $\longrightarrow$ применение sqrt к 25
   $\longleftarrow$ возвращаемый результат: 5.0

3.3 Задание 3

Функция, вычисляющая объём параллелепипеда по трём его сторонам:

Листинг 2 – Функция вычисления объёма параллелепипеда

```
1     (defun v (a b c) (* a b c))
```

Диаграмма вычисления функции:

$\Longrightarrow$ (v 2 3 4)
  $\Longrightarrow$ (* 2 3 4)
  $\longrightarrow$ применение * к 2, 3, 4
   $\longleftarrow$ возвращаемый результат: 24.0

3.4 Задание 4

Результаты вычисления выражений представлены в таблице 1. Значком * в номере выражения обозначается исправленное выражение, которое возможно вычислить.

Таблица 1 – Выражения и результаты их вычислений задания 4

№	Выражение	Результат
1	(list 'a c)	UNBOUND-VARIABLE c
2	(cons 'a (b c))	UNBOUND-VARIABLE c
2*	(cons 'a '(b c))	(a b c)
3	(cons 'a '(b c))	(a b c)
4	(caddy (1 2 3 4 5))	Illegal function call 1, caddy is undefined
4*	(caddr '(1 2 3 4 5))	3
5	(cons 'a'b'c)	Invalid number of arguments
5*	(cons 'a'b)	(a . b)
6	(list 'a (b c))	UNBOUND-VARIABLE c
6*	(list 'a '(b c))	(a (b c))
7	(list a '(b c))	UNBOUND-VARIABLE a
7*	(list 'a '(b c))	(a (b c))
8	(list (+ 1 '(length '(1 2 3))))	TYPE-ERROR
8*	(list (+ 1 (length '(1 2 3))))	(4)

3.5 Задание 5

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

Листинг 3 – Функция longer_then с использованием length

```
1    (defun longer_than (list1 list2) (> (length list1) (length list2)))
```

Листинг 4 – Функция longer_then с использованием базисных функций

```
1    (defun longer_than_2 (list1 list2)
2        (cond ((null list1) nil)
3              ((null list2) T)
4              (T (longer_than_2 (cdr list1) (cdr list2)))))
```

3.6 Задание 6

Результаты вычисления выражений представлены в таблице 2.

Таблица 2 – Выражения и результаты их вычислений задания 6

№	Выражение	Результат
1	(cons 3 (list 5 6))	(3 5 6)
2	(cons 3 ’(list 5 6))	(3 list 5 6)
3	(list 3 ’from 9 ’lives (- 9 3))	(3 from 9 lives 6)
4	(+ (length for 2 too) (car ’(21 22 23)))	UNBOUND-VARIABLE for
4*	(+ (length ’(for 2 too)) (car ’(21 22 23)))	24
5	(cdr ’(cons is short for ans))	(is short for ans)
6	(car (list one two))	UNBOUND-VARIABLE one
7 (6*)	(car (list ’one ’two))	one

3.7 Задание 7

Функция mystery представлена в листинге 5. Результаты вычисления выражений представлены в таблице 3.

Листинг 5 – Функция mystery

```
1     (defun mystery (x) (list (second x) (first x)))
```

Таблица 3 – Выражения и результаты их вычислений задания 6

№	Выражение	Результат
1	(mystery (one two))	UNBOUND-VARIABLE two, one is undefined
2	(mystery one ’two))	UNBOUND-VARIABLE one
(1,2)*	(mystery ’(one two))	(two one)
3	(mystery (last one two))	UNBOUND-VARIABLE one, two
3*	(mystery (last ’(one two)))	(nil two)
4	(mystery free)	UNBOUND-VARIABLE free -> free is not list
4*	(mystery ’(free))	(nil free)