|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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

**Отчет**

**по лабораторной работе № 3**

**Дисциплина: «Функциональное и логическое программирование»**

Выполнила: Овчинникова А. П.

Группа: ИУ7-65Б

Преподаватель: Толпинская Н.Б.

Москва, 2020

**Теоретическая часть.**

Список – это динамическая структура данных, которая может быть пустой или непустой. Если список не пуст, у него есть «голова» и «хвост», при этом «хвост» является списком.

Отличия Лиспа от других языков:

* Лисп ориентирован на обработку символов.
* Программа в Лисп представляет собой список. Данные также представляются в виде списка. Таким образом и программа, и данные хранятся в куче.

Классификация функций:

1. Чистые математические функции – это функции, которые имеют фиксированное количество аргументов и всегда возвращают один результат.
2. Форма – это функции, имеющие переменное количество аргументов, либо по-разному обрабатывающие свои аргументы.
3. Функционал – это функция, в качестве аргумента принимающая функцию.

Базис Лиспа – это атомы и структуры из простейших бинарных узлов, а также несколько базовых функций и функционалов. Базис содержит встроенные (примитивные) функции, которые анализируют, строят и разбирают любые структурные значения, и встроенные специальные функции и функционалы, которые управляют обработкой структур, представляющих вычисляемые выражения.

Классификация базисных функций:

1. Функции-селекторы: *car, cdr. Car, cdr* – чистые математические функции. Работают только со структурами.
2. Функции-конструкторы. Создают структуры.

*cons –* позволяет создавать списки. Возвращает бинарную ячейку. Требует два аргумента. В зависимости от второго аргумента возвращает точечную пару или список.

Существует также форма *list.* Она является более удобной формой функции *cons,* но не входит в базис. Создает столько списковых ячеек, сколько ей было передано аргументов. Всегда возвращает список.

1. Предикаты. Выполняют проверку.
   1. *atom –* является
   2. *null* – является ли список пустым.
   3. *Listp* – является ли структура списком.
   4. *Consp* – состоит ли структура из списковых ячеек.
   5. Функции сравнения:
      * *eq* – сравнивает по указателям. Указывает ли указатель на одно и то же.
      * *eql* – сравнивает по указателям (как *eq)* и проверяет числа одного «типа» (одного синтаксического представления).
      * *=* –если все аргументы(атомы) эквивалентны - возвращается *Т*, если хотя бы один не равно - *nil*. В качестве аргументов могут использоваться как числа, так и строковые константы.
      * *equal* – сравнивает как *eql* и корректно сравнивает списки.
      * *equalp* – может сравнивать объекты разной природы.

**Практическая часть.**

**Задание 1.**

1. *(equal 3 (abs -3))*

Вычисляется 3 к 3

Обработка функции *abs*

Вычисление -3 к -3

Применение *abs* к -3

3

Применение *equal* к 3 и 3

*T*

1. *(equal (+ 1 2) 3)*

*(+ 1 2)*

Вычисляется 1 к 1

Вычисляется 2 к 2

Функция *+* к 1 и 2

3

Вычисляется 3 к 3

Применение *equal* к 3 и 3

*T*

1. *(equal (\* 7 4) 21)*

*(\* 7 4)*

Вычисляется 7 к 7

Вычисляется 4 к 4

Функция *\** к 7 и 4

28

Вычисляется 21 к 21

Применение *equal* к 28 и 21

*Nil*

1. *(equal (\* 2 3) (+ 7 2))*

*(\* 2 3)*

Вычисляется 2 к 2

Вычисляется 3 к 3

Функция \* к 2 и 3

6

*(+ 7 2)*

Вычисляется 7 к 7

Вычисляется 2 к 2

Функция + к 7 и 2

9

Применение *equal* к 6 и 9

*Nil*

1. *(equal (- 7 3) (\* 3 2))*

*(- 7 3)*

Вычисляется 7 к 7

Вычисляется 3 к 3

Функция – к 7 и 3

4

*(\* 3 2)*

Вычисляется 3 к 3

Вычисляется 2 к 2

Функция \* к 3 и 2

6

Применение *equal* к 4 и 6

*Nil*

1. *(equal (abs (- 2 4)) 3))*

*(abs (- 2 4))*

*(- 2 4)*

Вычисляется 2 к 2

Вычисляется 4 к 4

Функция - к 2 и 4

-2

Функция *abs* к -2

2

Вычисляется 3 к 3

Применение *equal* к 2 и 3

*Nil*

**Задание 2.**

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

*(defun hyp (a b)   
 (sqrt (+ (\* a a) (\* b b)))  
)*

Диаграмма вычисления:

*(hyp (3 4))*

Вычисляется 3 к 3

Вычисляется 4 к 4

Функция *hyp* к 3 и 4

Создание переменной *a* со значением 3

Создание переменной *b* со значением 4

*(sqrt (+ (\* a a) (\* b b)))*

*(+ (\* a a) (\* b b))*

*(\* a a)*

Вычисляется *a* к 3

Вычисляется *a* к 3

Функция \* к 3 и 3

9

*(\* b b)*

Вычисляется *b* к 4

Вычисляется *b* к 4

Функция \* к 4 и 4

16

Функция + к 9 и 16

25

Функция *sqrt* к 25

5

Возврат 5

**Задание 3.**

Написать функцию, вычисляющую объем параллелепипеда по 3-м его сторонам, и составить диаграмму её вычисления.

*(defun parallelepiped (a b c)  
 (\* a b c))*

Диаграмма вычисления:

*(parallelepiped (1 2 3))*

Вычисляется 1 к 1

Вычисляется 2 к 2

Вычисляется 3 к 3

Функция *parallelepiped* к 1 2 и 3

Создание переменной *a* со значением 1

Создание переменной *b* со значением 2

Создание переменной *c* со значением 3

*(\* a b c)*

Вычисляется *a* к 1

Вычисляется *b* к 2

Вычисляется *c* к 3

Функция \* к 1 2 и 3

6

Возврат 6

**Задание 4.**

*(list ‘a ‘b ‘c) = (A B C)*

*(cons ‘a (b c)) = Error.* Функция *B* не определена.

*(cons ‘a ‘(b c)) = (A B C)*

*(caddr (1 2 3 4 5)) = Error.* 1 не является вызовом функции.

*(cons ‘a ‘b ‘c) = Error.* Слишком много аргументов.

*(list ‘a (b c)) = Error.* Функция *B* не определена.

*(list a ‘(b c)) = Error. A* не имеет значения.

*(list (+ 1 ‘(length ‘(1 2 3)))) = Error. ‘(length ‘(1 2 3)))* не является числом.