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 курсу «Функциональное и логическое программирование»

Тема Определение функций пользователя

Студент Динь Вьет Ань

Группа ИУ7И-64Б

Оценка (баллы) _____

Преподаватели Толпинская Н. Б., Строганов Ю. В.

Теоретические вопросы

1. Базис языка `Lisp`

Базис – это минимальный набор инструментов языка и стркутур данных, который позволяет решить любые задачи.

Базис Lisp :

– атомы и структуры (представляющиеся бинарными узлами);

– базовые (несколько) функций и функционалов: встроенные — примитивные функции (atom, eq, cons, car, cdr); специальные функции и функционалы (quote, cond, lambda, eval, apply, funcall).

Функцией называется правило, по которому каждому значению одного или нескольких аргументов ставится в соответствие конкретное значение результата.

Функционалом, или функцией высшего порядка называется функция, аргументом или результатом которой является другая функция.

2. Классификация функций языка `Lisp`

Функции в языке `Lisp`:

- Чистые математические функции (имеют фиксированное количество аргументов, сначала выяисляются все аргументы, а только потом к ним применяется функция);
- Рекурсивные функции (основной способ выполнения повторных вычислений);
- Специальные функции, или формы (могут принимать произвольное количество аргументов, или аргументы могут обрабатываться по-разному);
- Псевдофункции (создают «эффект», например, вывод на экран);
- Функции с вариантами значений, из которых выбирается одно;
- Функции высших порядков, или функционалы – функции, аргументом или результатом которых является другая функция (используются для построения синтаксически управляемых программ)

- Базисные функции — минимальный набор функций, позволяющих решить любую задачу.

Также базисные и функции ядра можно классифицировать с точки зрения действий.

1. Селекторы — переходят по соответствующему указателю списковой ячейки.
2. Конструкторы — создают структуры данных.
3. Предикаты — позволяют классифицировать или сравнивать структуры.

3. Способы создания функций

- С помощью lambda. После ключевого слова указывается лямбда-список и тело функции.

```
(lambda (x y) (+ x y))
```

  Для применения используются лямбда-выражения.

```
((lambda (x y) (+ x y)) 1 2)
```

- С помощью defun. Используется для неоднократного применения функции (в том числе рекурсивного вызова).

```
(defun sum (x y) (+ x y))
(sum 1 2)
```

4. Функции Car и Cdr, eq, eql, equal, equalp

- Функция car от одного аргумента возвращает первый элемент списка, являющегося значением её аргумента;
- Функция cdr возвращает хвост списка, являющегося значением её единственного аргумента (хвостом, или остатком списка является список без своего первого элемента);
- Функция eq корректно сравнивает два символьных атома. Так как атомы не дублирутюся для данного сеанса работы, то фактически сравниваются соответсвующие указатели; Возвращает T, когда:

1. значением одного из аргументов является атом, и одновременно;
2. значения аргументов равны (идентичны). В ином случае значением функции eq является NIL. (eq 'ab 'Ab) => T, но (eq 1 2) => NIL.

- Функция eql корректно сравнивает атомы и числа одинакового типа (синтетической формы записи). Например, (eql 1 1) вернет T, а (eql 1 1.0) – Nil, так как целое значение 1 и значение с плавающей точкой 1.0 являются представителями различных классов;
- Функция equal работает идентично eql, но в дополнение умеет корректно сравнивать списки (считая списки эквивалентными, если они рекурсивно, согласно тому же equal, имеют одинаковую структуру и содержимое; считая строки эквивалентными, если они содержат одинаковые знаки);
- Функция equalp корректно сравнивает любые S-выражения.

5. Назначение и отличие в работе Cons и List

Cons — функция от двух аргументов. Создает списковую ячейку и расставляет 2 указателя – на голову и на хвост – на входные аргументы.

List — функция от произвольного числа аргументов, при этом все они вычисляются. Строит новый список, первым элементом которого является значение первого аргумента, хвостом – значение второго аргумента.

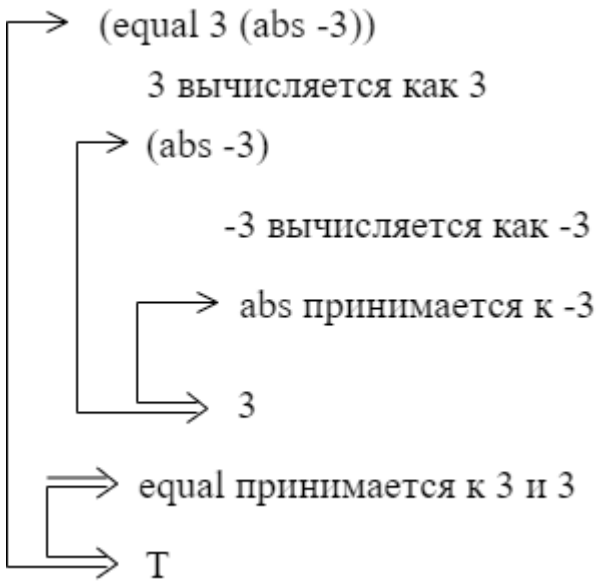
```
(cons '(A) '(B)) ;; ((A) B)
(list '(A) '(B)) ;; ((A) (B))
```

Практические задания

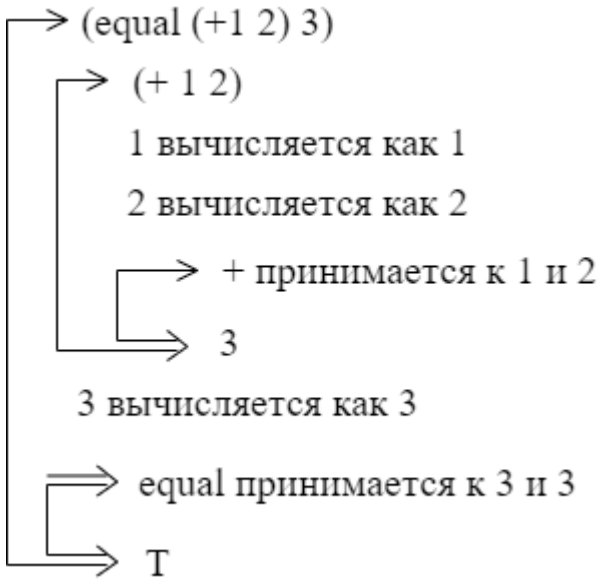
1. Задание 1

Составить диаграмму вычисления следующих выражений:

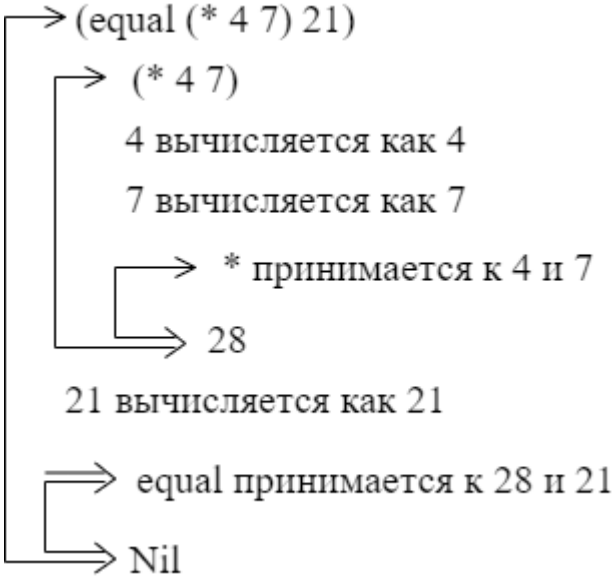
1. (equal 3 (abs -3))



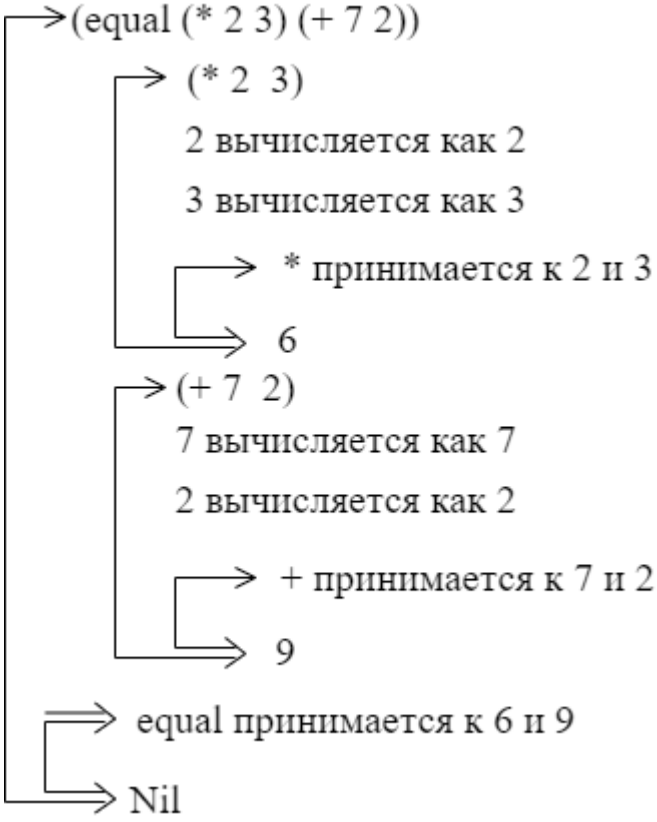
2. (equal (+ 1 2) 3)



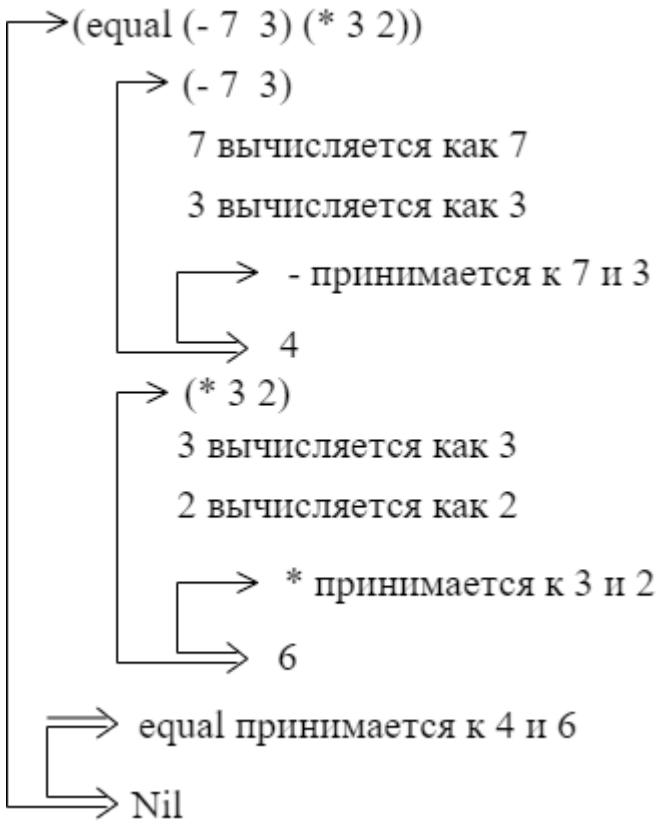
3. (equal (* 4 7) 21)



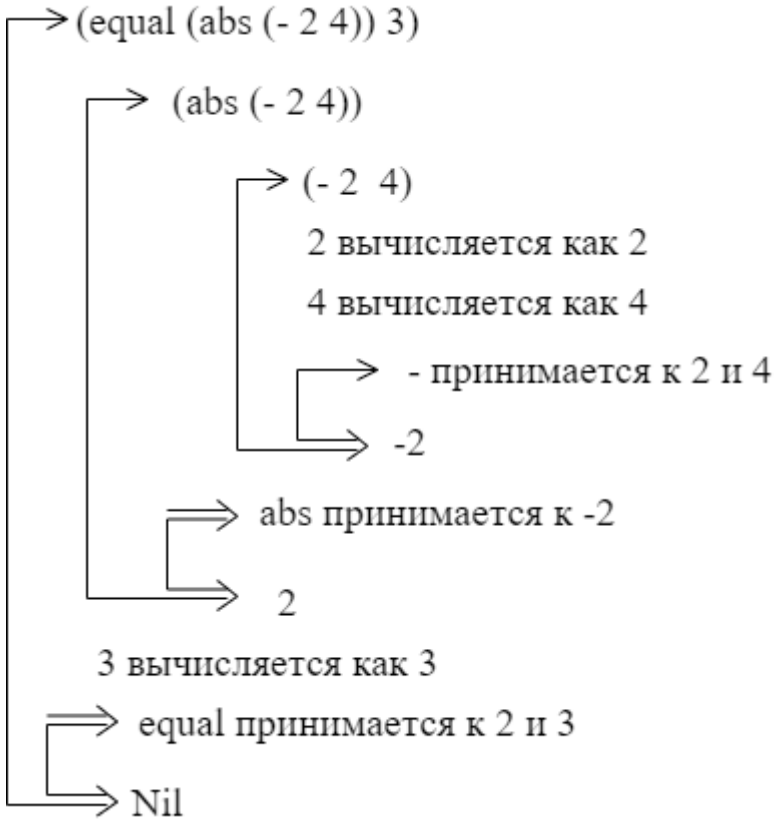
4. (equal (* 2 3) (+ 7 2))



5. (equal (- 7 3) (* 3 2))



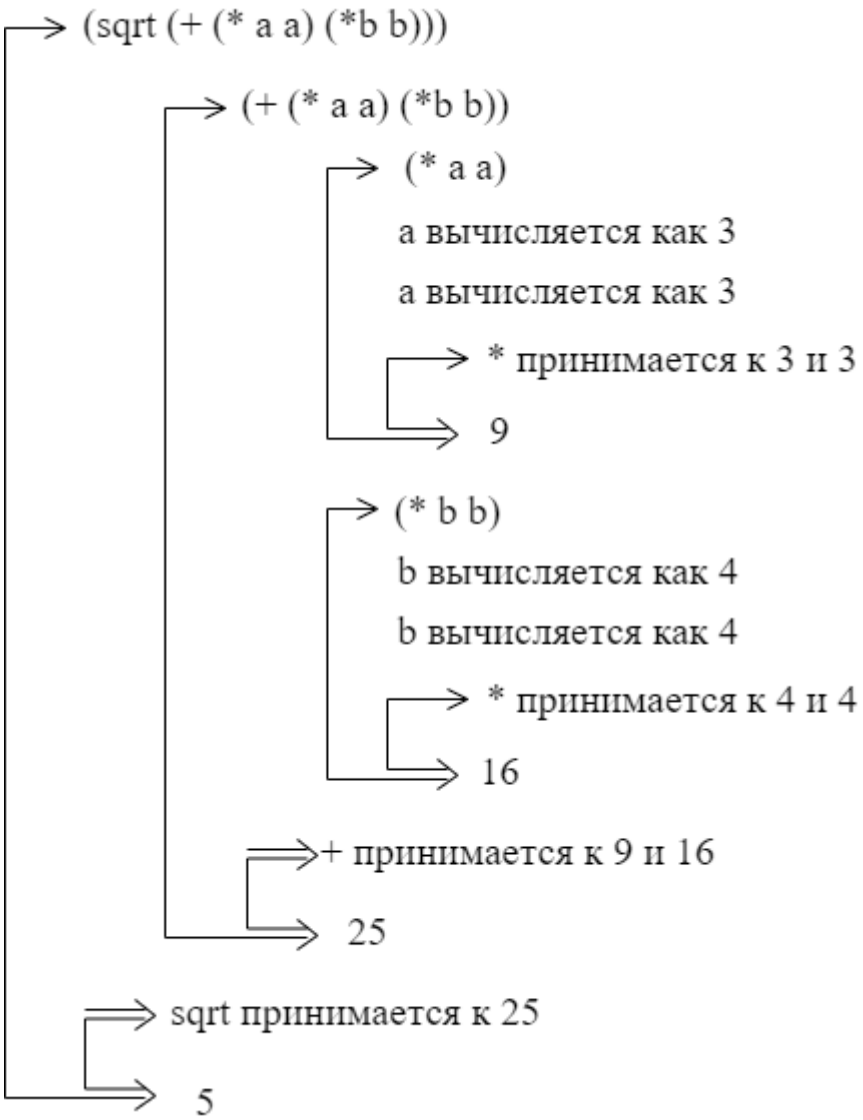
6. (equal (abs (- 2 4)) 3))



2. Задание 2

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

```
(defun hypotenuse (a b) (sqrt (+ (* a a) (* b b))))
(hypotenuse 3 4)  => 5.0
```



3. Задание 3

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

```
(list 'a c) ; The variable C is unbound.
=> (list 'a 'c)

(cons 'a (b c)) ; Undefined function : B, Undefined variable : C
=> (cons 'a '(b c))

(cons 'a '(b c)) => (A B C)

(caddr (1 2 3 4 5)) ;illegal function call
=> (caddr '(1 2 3 4 5))

(cons 'a 'b 'c)
;The function CONS is called with three arguments, but wants exactly two.
=> (list 'a 'b 'c)

(list 'a (b c)) ; Undefined function : B, Undefined variable : C
=> (list 'a '(b c))

(list a '(b c)) ; The variable A is unbound.
=> (list 'a '(b c))

(list (+ 1 '(length '(1 2 3))))
;The value (LENGTH '(1 2 3)) is not of type NUMBER
=> (list (+ 1 (length '(1 2 3))))
```

4. Задание 4

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

```
(defun longer_then (list1 list2) (> (length list 1) (length list2)))
```

5. Задание 5

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))  => (3 FROM 9 LIVES 6)
(+ (length for 2 too)) (car '(21 22 23))) ;The variable FOR is unbound.
(cdr '(cons is short for ans)) => (IS SHORT FOR ANS)
(car (list one two))     ;The variable ONE is unbound.
(car (list 'one 'two)) => ONE
```

6. Задание 6

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

```
(mystery (one two)) ;The variable TWO is unbound.
(mystery (last one two)) ;The variable ONE is unbound.
(mystery free) ;The variable FREE is unbound.
(mystery one 'two)) ;The variable ONE is unbound.
```

7. Задание 7

Написать функцию, которая переводит температуру в системе Фаренгейта температуру по Цельсию (defum f-to-c (temp)...).

```
(defun f_to_c (temp) (* (/ 5 9) (- temp 32.0)))
(f_to_c 451) => 232.77779
```

8. Задание 8

Что получится при вычисления каждого из выражений?

```
(list 'cons t NIL) => (CONS T NIL)
(eval (eval (list 'cons t NIL))) ;The function COMMON-LISP:T is undefined
(apply 'cons '(t NIL)) => (T)
(list 'eval NIL) => (EVAL NIL)
(eval (list 'cons t NIL))  => (T)
(eval NIL) => NIL
(eval (list 'eval NIL)) => NIL
```