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	ИУ7-64Б		Л.Е.Тартыков
	(Группа)	(Подпись, дата)	(И.О. Фамилия)
Преподаватель			Н.Б.Толпинская
		(Подпись, дата)	(И.О. Фамилия)
Преподаватель			Ю.В.Строганов
		(Подпись, дата)	(И.О. Фамилия)

Москва, 2022

1 Теоретические вопросы

1.1 Базис Lisp

Базис - минимальный набор конструкций языка и структур данных, с помощью которых можно решить любую задачу.

Базис Lisp образуют: атомы, структуры, базовые функции, базовые функционалы.

Базисные функции – минимальный набор функций, которые позволяют решить любую задачу.

1.2 Классификация функций

1. Чистые математические функции.

   Имеют фиксированное число аргументов и возвращают один результат. Сначала вычисляются все аргументы, затем к ним применяется исходная функция.

2. Специальные функции (формы).

   Специальные функции – функции, у которых переменное число аргументов или они обрабатываются по-разному (один вычисляется, другой - нет).

3. Псевдофункции.

   Псевдофункции – функции, которые создают "спецэффекты"; например, вывод на экран.

4. Функции с вариантами значений – выбирают какое-то одно значение.

5. Функционалы.

   Функционалы – функции, которые в качестве аргументов используют функции или возвращают в качестве результата функцию. Также они называются *функциями более высокого порядка*. Позволяют создавать

синтаксически управляемые программы (программы, которые сами создают какие-то функции; эти функции затем выполняются).

6. Рекурсивные.

1.3 Способы создания функций

1. lambda-выражение. Данный способ представлен с помощью формулы (1.1).

$$(lambda\ \lambda\text{-список форма}), \tag{1.1}$$

где λ-список – список формальных параметров, форма – тело функции.

lambda-выражение не хранится в памяти и не имеет имени. Вычисляется сразу же. Используется для повторных вычислений.

Вызов lambda-функции выполняется по формуле (1.2).

$$(\lambda\text{-выражение последовательность форм}) \tag{1.2}$$

2. С помощью *defun* по формуле (1.3).

$$(defun\ f\ \lambda\text{-выражение}) \tag{1.3}$$

Система по имени символьного атома находит его определение.

1.4 Функции Car и Cdr

car и *cdr* являются базовыми функциями доступа к данным.

car – принимает на вход один аргумент и возвращает первый элемент списка.

Пример использования car и списка представлен на листинге 1.1.

Листинг 1.1 – Использование car и списка.

```
1   (car '(A B C D))      ;; (A)
```

car – принимает на вход один аргумент и возвращает хвост списка.

Пример использования cdr и списка представлен на листинге 1.2.

Листинг 1.2 – Использование cdr и списка.

```
(cdr '(A B C D))      ;; (B C D)
```

1.5 Назначение и отличие в работе Cons и List

cons – имеет два аргумента и возвращает бинарный узел. Если вторым аргументом является атом, то возвращается точечная пара; если список – список.

list – имеет произвольное число аргументов и возвращает список.

Пример использования cons представлен на листинге 1.3.

Листинг 1.3 – Использование cons.

```
(cons 'A 'B)            ;; (A.B)
(cons 'A '(B C D))      ;; (A B C D)
(cons '(A B) '(C D))    ;; ((A B) C D)
```

Пример использования list представлен на листинге 1.4.

Листинг 1.4 – Использование list.

```
(list 'A 'B)            ;; (A B)
(list 'A '(B C) 'D)     ;; (A (B C) D)
```

функция list может быть представлена с помощью cons (листинг 1.5).

Листинг 1.5 – Представление list с помощью cons.

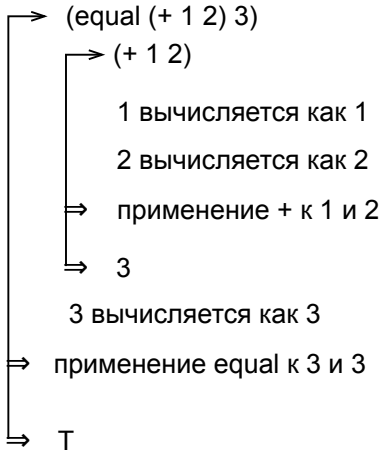
```
(defun list2 (ar1 ar2) (cons ar1 (cons ar2 ())))
(defun list3 (ar1 ar2 ar3) (cons ar1 (cons ar2 (cons ar3 ()))))
...
```

2 Практические задания

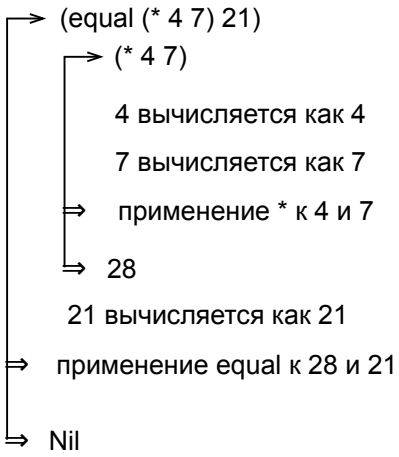
2.1 Составить диаграмму вычисления следующих выражений:

1. (equal 3 (abs -3))

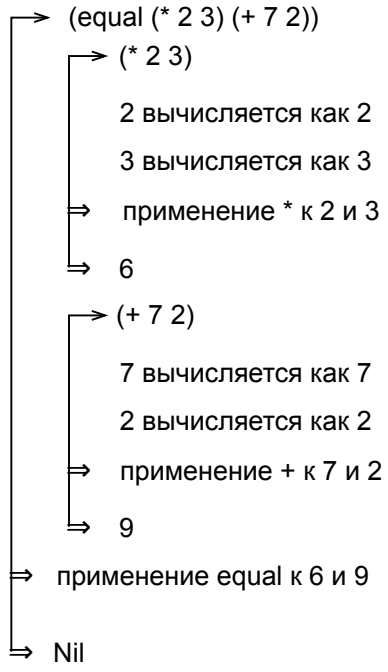
2. (equal (+ 1 2) 3)



3. (equal (* 4 7) 21)

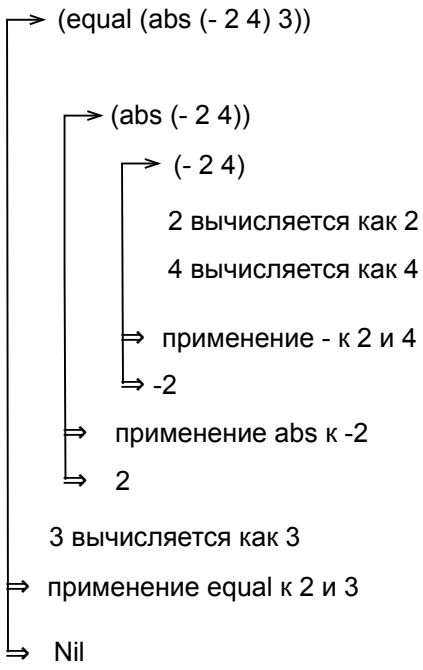


4. (equal (* 2 3) (+ 7 2))



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

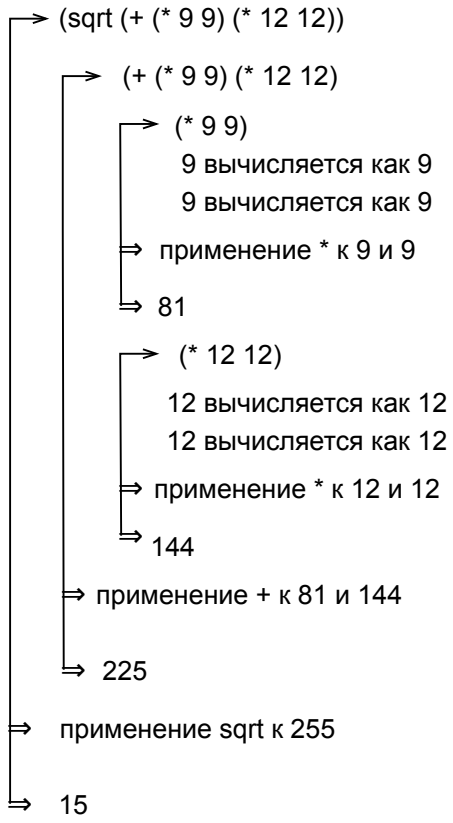
6. (equal (abs (- 2 4) 3))



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

Листинг 2.1 – Задание 2.

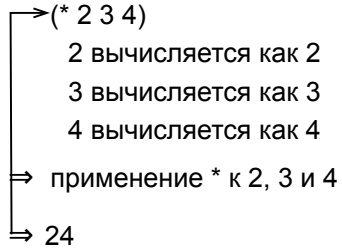
```
1 (defun hypotenuse (cath1 cath2)
2     (sqrt (+ (* cath1 cath1) (* cath2 cath2))))
```



2.3 Написать функцию, вычисляющую объем параллелепипеда по трем его сторонам, и составить диаграмму ее вычисления.

Листинг 2.2 – Задание 3.

```
1 (defun volume (side1 side2 side3) (* side1 side2 side3))
```



2.4 Каковы результаты вычисления следующих выражений? Объяснить возможную ошибку и варианты ее устранения

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

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

Листинг 2.3 – Задание 5.

```
1 (defun longer_then (arg1 arg2) (> (length arg1) (length arg2)))
```

2.6 Каковы результаты вычисления следующих выражений?

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)) ⇒ ошибка
5. (car '(21 22 23))) ⇒ ошибка
6. (cdr '(cons is short for ans)) ⇒ (is short for ans)
7. (car (list one two)) ⇒ ошибка
8. (car (list 'one 'two)) ⇒ one

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

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

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

Формулы:

1. $c = 5/9 * (f - 320)$;

Листинг 2.4 – Перевод из Фаренгейт в Цельсий.

```
1    (defun f_to_c (temp) (* (/ 5 9) (- temp 32.0)))
```

2. $f = 9/5 * c + 32.0$;

Листинг 2.5 – Перевод из Цельсия в Фаренгейт.

```
1    (defun c_to_f (temp) (+ (* (/ 9 5) temp) 32.0))
```

Как бы назывался роман Р.Брэдбери «+451 по Фаренгейту» в системе по Цельсию? «+232.778 по Цельсию»

2.9 Что получится при вычислении каждого из выражений?

1. (list 'cons t NIL) ⇒ (cons t Nil)
2. (eval (list 'cons t NIL))

   (eval (cons t Nil))

   (eval (t)) ⇒ ошибка
3. (eval (eval (list 'cons t NIL)))

   (eval (eval (cons t Nil)))

   (eval (eval (t))) ⇒ ошибка
4. (apply #cons "(t NIL)) ⇒ ошибка

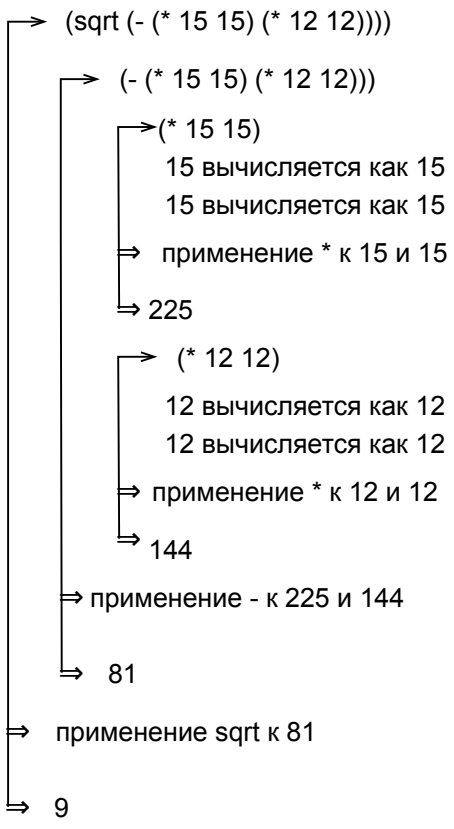
   (apply #'cons '(t NIL)) ⇒ (t)
5. (list 'eval NIL) ⇒ (eval NIL)
6. (eval NIL) ⇒ Nil
7. (eval (list 'eval NIL)) ⇒ Nil

Дополнительные задания

2.10 Написать функцию, вычисляющую катет по заданной гипотенузе и другому катету прямоугольного треугольника, составить диаграмму ее вычисления.

Листинг 2.6 – Дополнительное задание 1.

```
1    (defun hypo_cath (hypo cath)
2        (sqrt (- (* hypo hypo) (* cath cath))))
```



2.11 Написать функцию, вычисляющую площадь трапеции по ее основаниям и высоте, составить диаграмму ее вычисления.

Листинг 2.7 – Дополнительное задание 2.

```
1    (defun trapezoid_square (base1 base2 height)
2        (* 0.5 (+ base1 base2) height))
```