Государственное образовательное учреждение высшего профессионального образования

«Московский государственный технический университет имени Н. Э. Баумана»

(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Отчет по лабораторной работе №6

по курсу ФиЛП

Студент _____ Якубаускайте М.А.
(Подпись, дата)

Преподаватель _____ Толпинская Н.Б.
(Подпись, дата)

Москва 2019

Теоритическая часть.

Способы определения функции в LISP.

а) Макроопределение DEFUN - форма, которая особым образом обрабатывает некоторые из своих аргументов (DEFUN имя (список аргументов) (тело));

б) λ выражение - способ записи выражения, когда вначале записывается формула, которую надо вычислить, а затем значения, над которыми её вычислить.
(LAMBDA (аргументы) (тело))

1) "тяжеловесный"способ использования λ - выражений, возник из лямбда-исчисления Чёрча из математики.
((LAMBDA (x,y) (+ x y)) 2 5) - дает 7;
2) использование APPLY особой функции называемой функционалом. Требуется #.
(APPLY #'name (параметры))
(APPLY #'(LAMBDA (x y) (+ x y)) (2 5)). λ-описание часто используется при работе со структурированными списками, обходить которые нужно рекурсивно.

Вызов функции и блокировка.

Вызов функции производится с помощью функционала APPLY - функционал, который применяет аргумент к остальным аргументам - приминяющий функционал. Принимает ровно 2 аргумента - функциональный и список произвольной длины. Применяет функциональный элемент к списку, что приводит к вычислению функции, заданной первым аргументом, со списком параметров, заданным вторым аргументом. Происходит «вызов функции».
#′ - «function name» - reader macros - результатом вычисления является функция-объект, к которой можно применить APPLY или FUNCALL.
Функционалы - особые функции, формы, которые в качестве аргумента принимают другие функции (применяющие и отображающие).
- функциональная блокировка
' - блокировка вычислений
Функциональная блокировка FUNCTION. С её помощью можно зафиксировать контекст определения функции. В случае если функция не имеет свободных переменных, эти два варианта блокирования не отличаются в процессе обработки. (FUNCALL #'FUN ARG1 ARG2 ...) - FUN можно вычислять. Поскольку вычислениям подвергаются символьные выражения, этот функционал позволяет динамично строить выражения функции, применяя её к указанным аргументам. Единственное требование - после вычисления первого аргумента выражение должно представлять собой функ-

ции. Результат обработки не может быть макросом. FUNCALL является противоположностью FUNCTION.

Глобальные и локальные символьные атомы.

Lisp - интерпритатор в ходе своей работы поддерживаеь специальную таблицу символьных атомов - таблицу символов, в которой хранится информация обо всех атомах, встретившихся в тексте интерпритируемой программы или используемых в качестве обрабатываемых данных. Для присваивания рабочей переменной значения применяется форма SET. Чтобы присвоить переменной pi значение 22/7:

```
1 или
2 \begin{lstlisting} (SETQ PI 22/7)
```

В общем случае атом имеет несколько разных значений. Они независимы друг от друга, один и тот же символьный атом может быть использован как имя функции и как имя формального параметра. Нужная его интерпретация обеспечивается, исходя из контекста его применения. В таблице символьных атомов для каждого атома хранятся не сами значения, а указатели на внутренние представления этих значений. Внутренним представлением символьного атома является указатель на соответсвующий элемент таблицы атомов.

Глобальные атомы определяются в глобальной области видимости. Локальные атомы определяются в области видимости функции.

Практическая часть.

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

```
1 (DEFUN C (A B) (SQRT (+ A A) (*B B))))
```

3.1 - Написать функцию, которая принимает целое число и возвращает первое четное число, не меньшее аргумента.

```
1 (DEFUN F(X) (IF (ODD X) (+ X) (+ X 2)))
```

3.2 - Написать функцию, которая принимает число и возвращает число, того же знака, но с модулем на 1 больше аргумента.

```
1 (DEFUN F(X) (IF (> X 0) (+ X 1) (- X 1)))
```

3.3 - Написать функцию, которая принимает два числа и возвращает список из этих чисел, расположенный по возрастанию.

```
1 (DEFUN F(X Y) (IF (< X Y) (LIST X Y) (LIST Y X)))
```

3.4 - Написать функцию, которая принимает три числа и возвращает T только тогда, когда первое число расположено между первым и вторым.

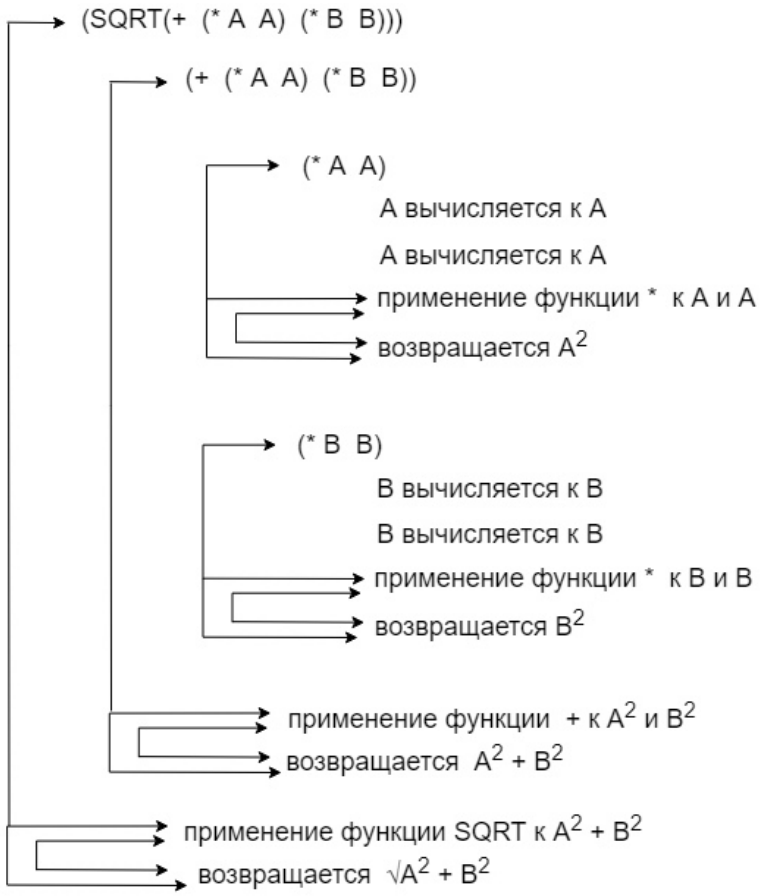

Рисунок 0.1 — Диаграмма функции 2.2.

```
1 (DEFUN F(X Y Z) (OR (AND (> X Y) (< X Z) (AND (> X Z) (< X Y)))))
```

2.7 - Написать функцию, которая переводит температуру в системе Фаренгейта в температуру по Цельсию. Как бы назывался роман Р. Брэдбери "+451 по Фаренгейту"в системе по Цельсию?

```
1 (DEFUN F_TO_C (TEMP) (/ 5 9) (- TEMP 320)))
```

```
1 (F\_TO\_C 451) => 72.77778
```

2.8 - Что получится при вычислении выражений?

```
1 (LIST 'CONS T NIL) => (CONS T NIL)
2 (EVAL (EVAL (LIST 'CONS T NIL))) => UNDERFINED FUNCTION T
3 (APPLY #'CONS '(T NIL)) => (T)
4 (LIST 'EVAL NIL) => (EVAL NIL)
5 (EVAL (LIST 'CONS T NIL)) => (T)
6 (EVAL NIL) => NIL
7 (EVAL (LIST 'EVAL NIL)) => NIL
```