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, искусственный интелект и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Отчёт по лабораторной работе № 1 по курсу «Функциональное и логическое программирование»

Тема Списки в Lispe. Использование стандартных функций

Студент Калашков П. А.

Группа ИУ7-66Б

Оценка (баллы)

Преподаватель Толпинская Н. Б.

Москва — 2023 г.

1 Теоретические вопросы

Вопрос 1. Элементы языка: определение, синтаксис, представление в памяти

Элементы языка и их определение

Вся информация (данные и программмы) в Lisp представляется в виде символных выражений — S-выражений. По определению:

S-выражение := <атом>|<точечная пара>

Элементами языка Lisp являются атомы и точечные пары (структуры). К атомам относятся:

1) символы (идентификаторы) — набор литер, начинающихся с буквы;

2) специальные символы для обозначения логических констант T, Nil;

3) самоопределимые атомы — натуральные числа, дробные числа, вещественные числа, строки (последовательность символов, заключённых в двойные апострофы).

Точечные пары ::= (<атом>, <атом>) |
(<атом>, <точечная пара>) |
(<точечная пара>, <атом>) |
(<точечная пара>, <точечная пара>)
Список ::= <пустой список> | <непустой список>), где
<пустой список> ::= () | Nil,
<непустой список> ::= (<первый элемент>, <хвост>),
<первый элемент> ::= (S-выражение),
<хвост> ::= <список>

Синтаксис элементов языка и их представление в памяти

Синтаксически любая структура (точечная пара или список) заключается в `()`:
`(A . B)` – точечная пара
`(A)` – список из одного элемента
Пустой список изображается как `Nil` или `()`
Непустой список может быть изображён: `(A. (B . (C ()))` или `(A B C)`
Элементы списка могут являться списками: `((A) (B) (C))`
Любая непустая структура `Lisp` в памяти представлена списковой ячейкой, хранящей два указателя: на голову (первый элемент) и хвост (всё остальное).

Вопрос 2. Особенности языка Lisp. Структура программы. Символ апостроф

Lisp — интерпретируемый символьный язык программирования, т.е. язык программирования, предназначенный для символных вычислений и преобразований, символьной обработки.

Программа и данные в Lisp представлены списками, поскольку программа — текст, и синтаксического различия между программой и данными нет. Это даёт возможность выдать программу за данные и заставить её менять саму себя. По умолчанию список считается вычислимой формой, в которой первый элемент — название функции, остальные элементы - аргументы функции.

В основе языка Lisp лежит λ-исчисление, согласно которому, любые вычислительные выражения могут быть преобразованы в вид функций от 1-го аргумента.

Поскольку программа и данные представлены списками, то их нужно както различать. Для этого была создана функция quote, а ’ — ее сокращенное обозначение. quote — функция, блокирующая вычисление.

Таким образом, символ апострофа ’ – функциональная блокировка, эквивалентен функции `quote`. Блокирует вычисление выражения. Таким образом,

выражение воспринимается интерпретатором как данные.

Вопрос 3. Базис языка Lisp. Ядро языка

Базис языка представлен:

1) атомами;

2) структурами;

3) функциями:
   `atom, eq, cons, car, cdr,`
   `cond, quote, lambda, eval, label`.

2 Практические задания

Задание 1

Представить следующие списки в виде списочных ячеек:

1) `'(open close halph)`

2) `'((open1) (close2) (halph3))`

3) `'((one) for all (and (me (for you))))`

4) `'((TOOL) (call))`

5) `'((TOOL1) ((call2)) ((sell)))`

6) `'(((TOOL) (call)) ((sell)))`

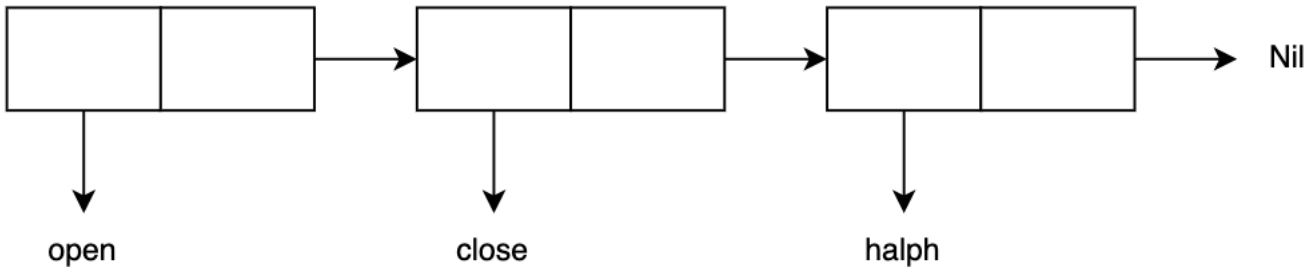

Рисунок 2.1 – `'(open close halph)`

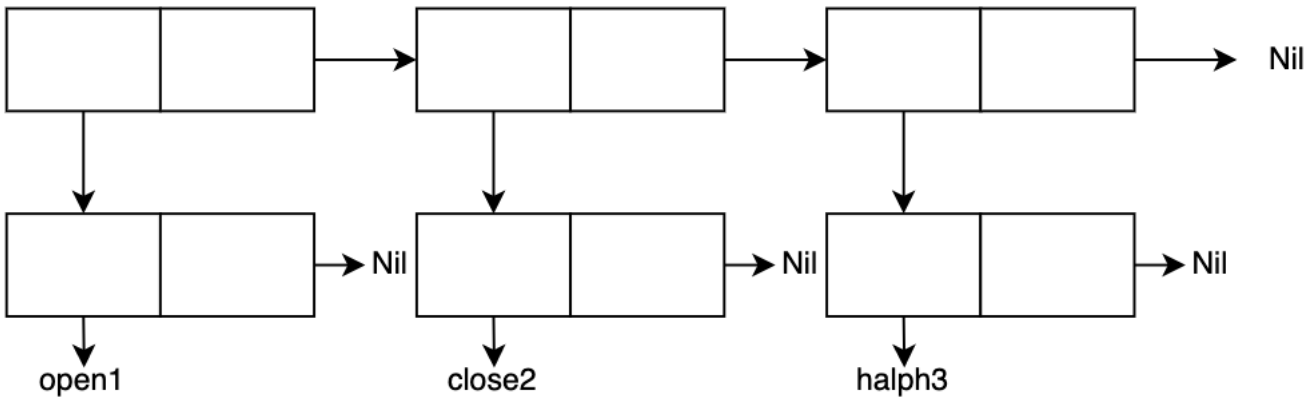

Рисунок 2.2 – `'((open1) (close2) (halph3))`

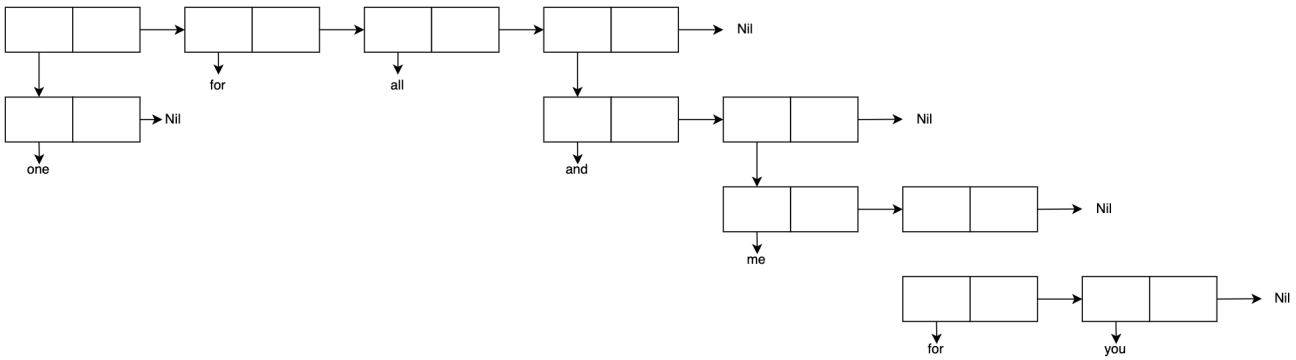

Рисунок 2.3 – `'((one) for all (and (me (for you))))`

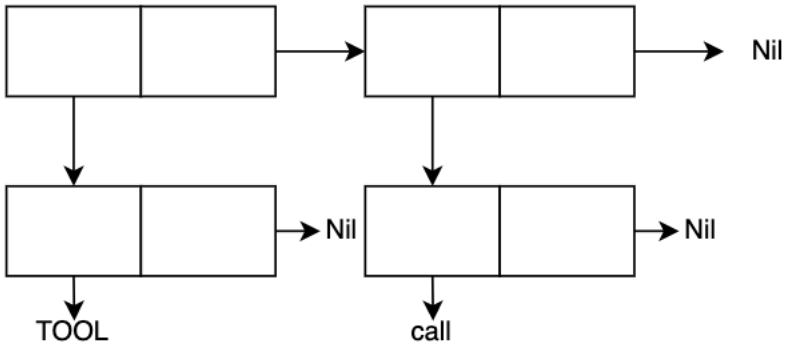

Рисунок 2.4 – `'((TOOL) (call))`

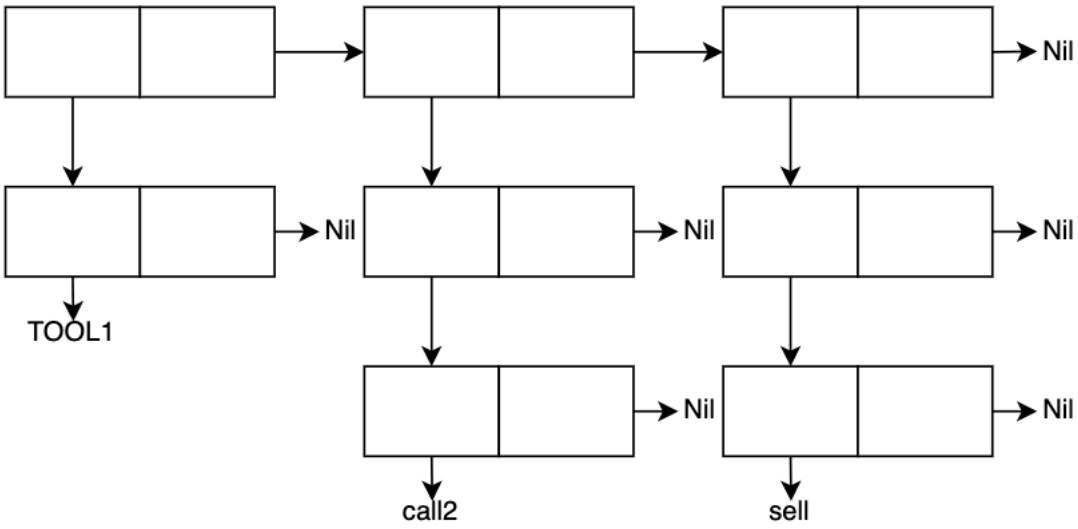

Рисунок 2.5 – `'((TOOL1) ((call2)) ((sell)))`

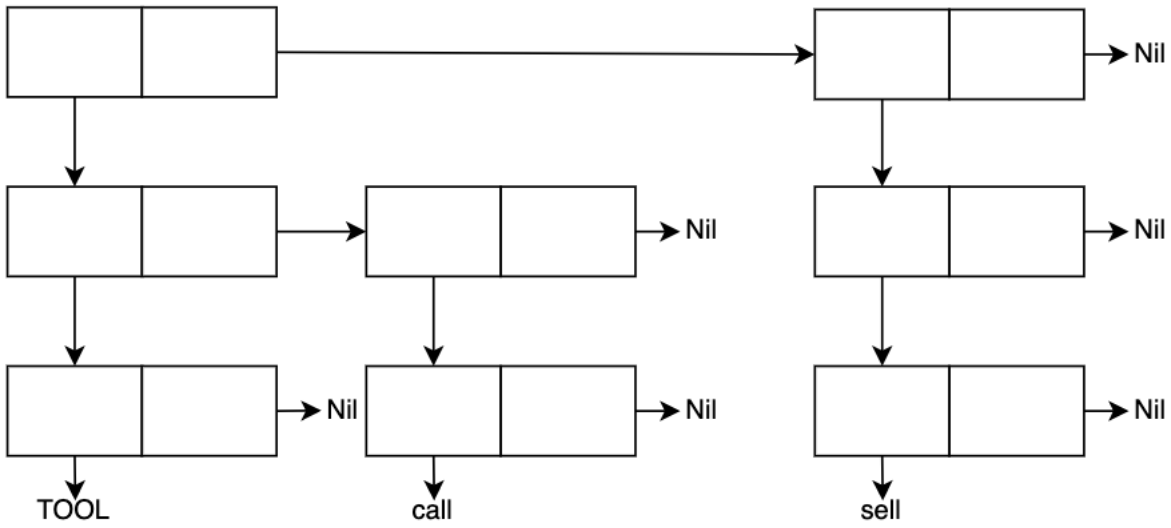

Рисунок 2.6 – `'(((TOOL) (call)) ((sell)))`

Задание 2

Используя только функции `CAR` и `CDR`, написать выражения, возвращающие:

1) второй;

2) третий;

3) четвёртый;

элементы заданного списка.

Решение:

1) второй `(car (cdr '(1 2 3 4 5)))`;

2) третий `(car (cdr (cdr '(1 2 3 4 5))))`;

3) четвёртый `(car (cdr (cdr (cdr '(1 2 3 4 5)))))`.

Задание 3

Что будет в результате вычисления выражений?

a) `(CAADR '((blue cube) (red pyramid)))` // red

b) `(CDAR '((abc) (def) (ghi)))` // Nil

c) `(CADR '((abc) (def) (ghi)))` // (def)

d) `(CADDR '((abc) (def) (ghi)))` // (ghi)

Задание 4

Напишите результат вычисления выражений и объясните как он получен:

Листинг 2.1 – Задание 4

```
(list 'Fred 'and 'Wilma) ; (fred and wilma)
(list 'Fred '(and Wilma)) ; -> (fred (and wilma))
(cons Nil Nil) ; -> (Nil)
(cons T Nil) ; -> T
(cons Nil T) ; -> (Nil . T)
(list Nil) ; -> (Nil)
(cons '(T) Nil) ; -> ((T))
(list '(one two) '(free temp)) ; -> ((one two) (free temp))
(cons 'Fred '(and Wilma)) ; -> (Fred and Wilma)
(cons 'Fred '(Wilma)) ; -> (Fred Wilma)
(list Nil Nil) ; -> (Nil Nil)
(list T Nil) ; -> (T Nil)
(list Nil T) ; -> (Nil T)
(cons T (list Nil)) ; -> (T Nil)
(list '(T) Nil) ; -> ((T) Nil)
(cons '(one two) '(free temp)) ; -> ((one two) free temp)
```

Задание 5

Написать λ-выражение и соответствующую функцию.

— написать функцию `(f ar1 ar2 ar3 ar4)`, возвращающую список `((ar1 ar2) (ar3 ar4))`;

— написать функцию `(f ar1 ar2)`, возвращающую список `((ar1) (ar2))`;

— `(f ar1)`, возвращающую список `(((ar1)))`.

Листинг 2.2 – Задание 5

```
(defun f1 (ar1 ar2 ar3 ar4)
    (list (list ar1 ar2) (list ar3 ar4)))

(defun f2 (ar1 ar2)
    (list (list ar1) (list ar2)))

(defun f3 (ar1)
  (list (list (list ar1))))
```

Результаты в виде списочных ячеек представлены на рисунках

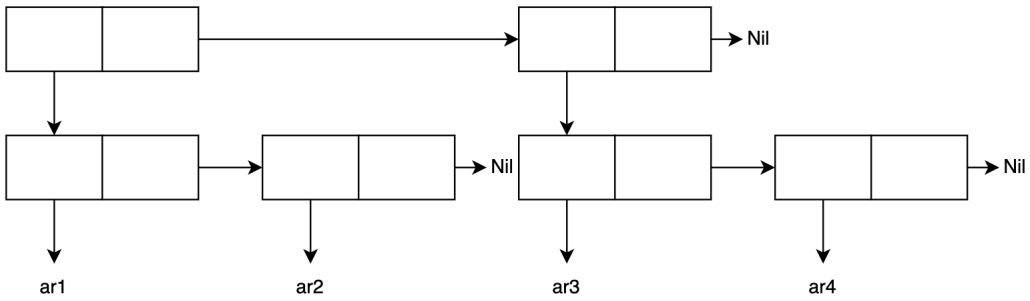

Рисунок 2.7 – `((ar1 ar2) (ar3 ar4))`

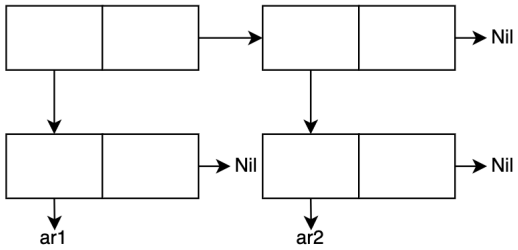

Рисунок 2.8 – `((ar1) (ar2))`

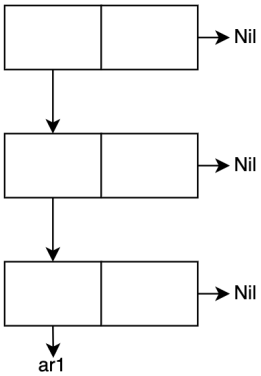

Рисунок 2.9 – `(((ar1)))`