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»

Лабораторная работа №2

Студент: Луговой Д.М.
Группа: ИУ7-61Б
Преподаватель: Толпинская Н.Б.

Москва, 2020 г.

Цель работы: приобрести навыки использования списков и стандартных функций Lisp.

Задачи работы: изучить способ использования списков для фиксации информации, внутреннее представление одноуровневых и структурированных списков, методы их обработки с использованием базовых функций Lisp.

1. **Базис LISP**

   Базис языка – это необходимый минимальный набор конструкций, которые должные обязательно присутствовать, чтобы при их помощи составлять команды. Базис языка Lisp образуют атомы, структуры, базовые функции и функционалы.

2. **Классификация функций**

   Функция есть однозначное отображение множества исходных данных на множество её значений. Функции классифицируются на:

   - Чистые математические функции - принимают фиксированное количество аргументов
   - Формы – принимают не фиксированное количество аргументов или обрабатывают аргументы по разному
   - Функционалы (высших порядков) – используют другие функции в качестве аргументов или вырабатывают в качестве результатов.
   - Базисные функции
     - Селекторы (функции доступа) - car, cdr
     - Конструкторы (функции создания структур) - cons, list
     - Предикаты - atom, null, listp, consp
     - Функции сравнения - eq, eql, equal, =, equalp

3. **Представление списков в памяти**

   Список – это структура данных. Может быть пустой и непустой. В памяти представлен списочной ячейкой, содержащей указатели на голову списка и на его хвост. Голова списка представляет из себя S-выражение, хвост является списком. Список может быть пустым. В Lisp возможны два типа представления пустого списка: пара пустых скобок и специальный символ Nil.

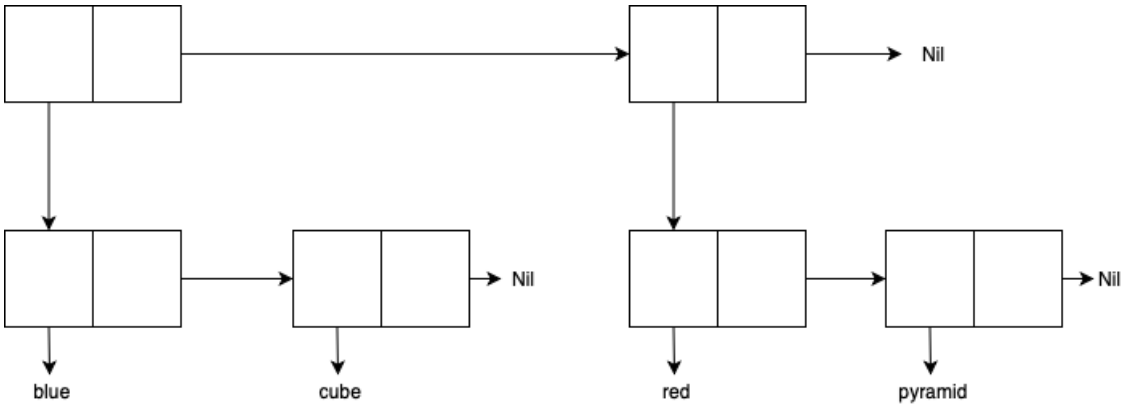
4. **Функции CAR и CDR**

   CAR и CDR являются базовыми функциями доступа к данным. CAR принимает точечную пару или пустой список в качестве аргумента и возвращает первый элемент или nil, соответственно. CDR принимает точечную пару или пустой список и возвращает список состоящий из всех элементов, кроме первого. Если в списке меньше двух элементов, то возвращается Nil.

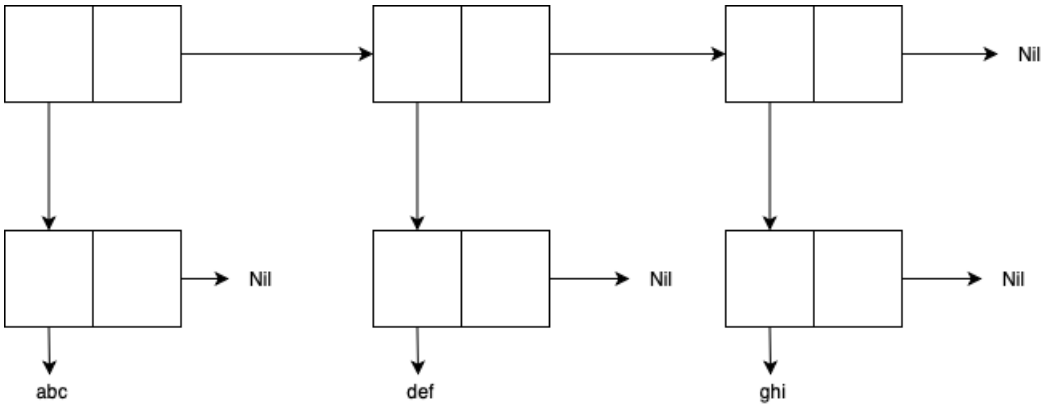
Задание №1

Что будет в результате вычисления выражений?

1. (CAADR ‘((blue cube) (red pyramid)))
   Результат: red



2. (CDAR ‘((abc) (def) (ghi)))
   Результат: Nil



3. (CADR ‘((abc) (def) (ghi)))
   Результат: (def)
4. (CADDR ‘((abc) (def) (ghi)))
   Результат: (ghi)

Задание №2

Напишите результат вычисления выражений:

1. (list 'Fred 'and Wilma)
   Результат: The variable WILMA is unbound. Wilma не может быть вычислено.
2. (list 'Fred '(and Wilma))
   Результат: (Fred (and Wilma))
3. (cons Nil Nil)
   Результат: (NIL)
4. (cons T Nil)
   Результат: (T)
5. (cons Nil T)
   Результат: (Nil . T)
6. (list Nil)
   Результат: (Nil)
7. (cons (T) Nil)
   Результат: The function COMMON-LISP: T is undefined. T не является функцией.
8. (list `(one two) `(free temp))
   Результат: ((one two) (free temp))
9. ((cons 'Fred '(and Wilma))
   Результат: (Fred and Wilma)
10. (cons `Fred `(Wilma))
    Результат: (Fred Wilma)
11. (list Nil Nil)
    Результат: (Nil Nil)
12. (list T Nil)
    Результат: (T Nil)
13. (list Nil T)
    Результат: (Nil T)

14. (cons T (list Nil))
    Результат: (T Nil)

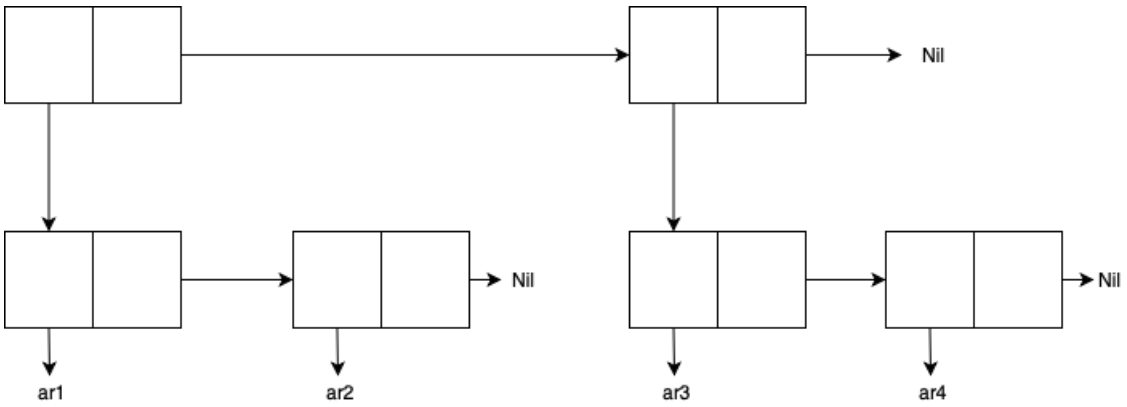
15. (list (T) Nil)
    Результат: The function COMMON-LISP: T is undefined. T не является функцией.

16. (cons ‘(one two) ‘(free tmp))
    Результат: ((one two) free tmp)

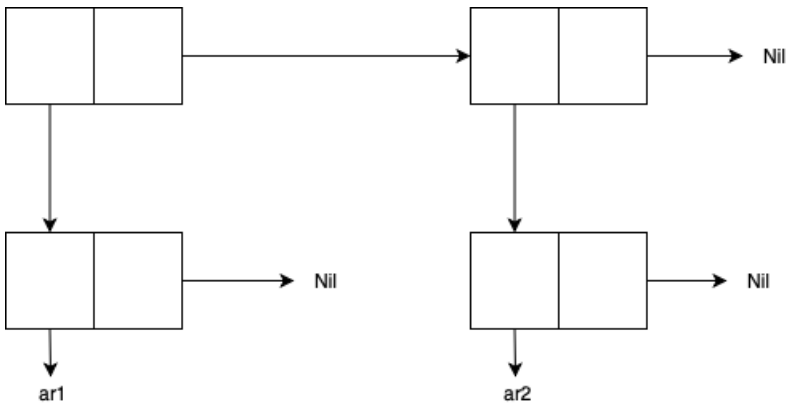
Задание №3

Написать функцию

1. (f ar1 ar2 ar3 ar4), возвращающую ((ar1 ar2) (ar3 ar4)):
   - (defun f (ar1 ar2 ar3 ar4) (list (list ar1 ar2) (list ar3 ar4)))
   - (defun f (ar1 ar2 ar3 ar4) ‘((,ar1 ,ar2) (,ar3 ,ar4)))
   - (defun f (ar1 ar2 ar3 ar4) (cons (cons 1 (cons 2 NIl))(cons (cons 3 (cons 4 Nil)) Nil)))



2. (f ar1 ar2), возвращающую ((ar1) (ar2)):
   Функция: (defun f (ar1 ar2) (list (list ar1) (list ar2)))



3. (f ar1), возвращающую (((ar1))):
   Функция: (defun f (ar1) (list (list (list ar1))))