МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ Н. Э. БАУМАНА

Факультет информатики и систем управления

Кафедра теоретической информатики и компьютерных технологий

Курсовой проект

по курсу «Конструирование компиляторов»

«Syntax sugar for Scheme»

Выполнил:
студент ИУ9-101
Выборнов А. И.

Руководитель:
Дубанов А. В.

Москва 2015

Содержание

- уточнить название работы
-
-

Введение

Lisp — это семейство динамических функциональных языков программирования. Первая версия языка Lisp была создана в 1958 году в ходе работ по созданию искусственного интеллекта. К настоящему времени сфера применения Lisp значительно увеличилась, а Lisp представляет собой целое семейство языков: Common Lisp, Scheme, Racket и другие.

В рамках работы рассматривается язык Scheme. Этот язык был разработан специально для учебных целей, благодаря чему включает в себя очень ограниченный, но весьма гибкий набор примитивов. Scheme удобен для написания скриптов и расширений (имеется специально для этого предназначенная реализация — GNU Guile).

Форматирование кода на Scheme, отслеживание парных открывающих и закрывающих скобок и некоторые другие синтаксические особенности требуют применения не слишком распространенных и привычных приложений таких как специализированная среда разработки, к примеру — Racket или текстовый реадактор со специальными расширениями, наиболее часто используется Emacs.

К числу недостатков можно также отнести: обилие скобок, затрудняющее чтение программы, отсутствие возможности записи выражений в привычном инфиксном формате.

Целью данной работы является разработка и реализация функционального динамического языка на основе Scheme, который имеет более дружелюбный синтаксис.

1. Теоретическая часть

1.1. Lisp

Поподробнее про лисп.

1.2. Функционал входного языка

вступленние программа представляет собой список из определений функций и их вызовов

мб ввести main и оставить только определение функций?

1.2.1. символы

про косяк в документации про строки как в лиспе, за исключением... примеры

1.2.2. функции

1)sign x | x > 0 = 1 (multiwayif)
| x == 0 = 0
| x < 0 = -1 2) sign 0 = 0 sign a = if a > 0 then 1 else -1
пример на Algebraic data type ?! я не работаю с типами :(
3)
x y -> x+y
какой должен быть вызов функции? мб сделать lisp-style?

1.2.3. условия

if a then b else c

1.2.4. инфиксная арифметика

при чём тут guile?

1.2.5. определение функций на scheme

мб вида: sign x: scheme (scheme body) ?

2. Объекты и методы

Характеристики программного обеспечения:

- Операционная система — Ubuntu 14.04 LTS x64.
- Язык программирования — Python 2.7.3.

Характеристики оборудования:

- Процессор — Intel Core i7-3770K 3.50 Гц 8 ядер.
- Оперативная память — 16 Гбайт DDR3.

3. Реализация

3.1. Используемые технологии

3.1.1. ANTLR

что круто, а что не очень обязательно про минусы python версии особенно про различное поведение при компиляции из файла и строки

3.1.2. graphviz

...

3.1.3. тестирование

...

3.1.4. сборка пакета

...

3.1.5. Детали реализации

общая структура:
$-> antlr syntax tree -> AST -> lisp_tree -> lisp_file$
особоенности каждого этапа преобразования

3.1.6. Обработка ошибок

сделано

3.1.7. Видимость символов

увы нет, надо обязательно доделать

3.2. Интерфейс

CLI все дела
примеры использования

4. Тестирование

что писать?

куда вставить сравнение синтаксиса с другими языками?

5. Заключение

Что получилось. Привнесена няшность, но возможности далеко не такие как были.

Список литературы

[1] стандарт r5rs

[2] ахо ульман книга дракона

[3] какая нибудь дока по antlr

[4] Jeffrey Dean, Sanjay Ghemawat. MapReduce: Simplified Data Processing on Large Clusters // Google, Inc. — 2004.

[5] MapReduce // slideshare: URL: http://www.slideshare.net/yandex/mapreduce-12321523

[6] What is the most efficient way to serialize in Python? // Quora: URL: http://www.quora.com/What-is-the-most-efficient-way-to-serialize-in-Python

[7] Hadoop save the World? // CODE1NSTINCT: URL: http://www.codeinstinct.pro/2012/08/hadoop-design.html

[8] MapReduce Patterns, Algorithms, and Use Cases // Highly Scalable Blog: URL: https://highlyscalable.wordpress.com/2012/02/01/mapreduce-patterns/