Compiler Construction: Practical Introduction

Project presentation

5 = 5 team

Fall Semester week 3 2023 Innopolis University

Main points

S Special forms Each special form is actually a list where the very first element is a keyword.

P Predefined functions All predefined functions perform some actions on their arguments and return some result.

A Arithmetic functions Have two parameters and returns some mathematically correct result.

O Operations on lists Do some basic list functions such as `head`, `tail` or `cons`.

C Comparisons Performs usual comparisons and return a boolean value depending on the result.

P Predicates The functions return a boolean value: `true`, if the argument is of type that the function expects, and `false` otherwise.

L Logical operators The functions perform usual logical operators on evaluated arguments and return a boolean value.

E Evaluator If the argument if a list then the function treats it as a valid program and tries to evaluate it. In that case, the function returns the value that the program issues. If the argument is a literal or atom the function just return the argument.

Grammar

The language grammar follows the ideas of its predecessors and is remarkably simple.

- Program : Element { Element }
- List : **(** Element { Element } **)**
- Element : Atom | Literal | List
- Atom : Identifier
- Literal : [+|-] Integer | [+|-] Real | Boolean | **`null`**
- Identifier : Letter { Letter | DecimalDigit }
- Letter : *Any Unicode character that represents a letter*
- Integer : DecimalDigit { DecimalDigit }
- DecimalDigit : **`0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9`**
- Real : Integer . Integer
- Boolean : **`true`** | **`false`**

Lexer

Some points we implemented in our lexer

1. Predefined functions are interpreted as indentifier so as not to load the lexer
2. We recognize arithmetic pluses and minuses as an int token
3. About the comments, we decided that we cut them out at the stage of processing the file with a lexer