Advance Programming Languages
Project Proposal
Fall 2014

The objective of the project is implementation of an interpreter for a functional language. This includes defining the syntax and semantic specifications of the language through the use of grammar and BNF notations. The language that is to be defined contains the following features

- Basic arithmetic functions
- Basic types of integer and boolean
- Boolean connectives and, or, not
- Relational operations
- Conditional if … else if … else statement
- Procedural definitions
- Type Checking
- Recursion
- Classes and Objects
- Method Overloading in Classes and Objects
- Constructor Overloading is also possible

Grammar:

The BNF notation for the grammar is as follows,

<program> ::={ <class-expression>}* <expression>

<class-expression> ::= class <class-identifier> ({members <identifier>}* {<method-decl>}*);

<method-decl> ::= method <identifier> {{<identifier>}*(,) <expression>}

<class-identifier> ::=<identifier>

<expression> ::= <number>

| <identifier>

| <primitive> ({<expression>}*(,))

<expression> ::= if <expression>

then <expression> else <expression

<expression> ::= let {<identifier> = <expression>} *

in <expression>

<expression> ::= cond {<expression> ==> <expression>}∗ end

<expression> ::= new <identifier> (<expression>*(,))

<expression> ::= send <identifier> (<expression>*(,))

<expression> ::=set <identifier> = <expression>

<expression>::= begin <expression> {; <expression>}* end

<expression> ::= letrec {<identifier>}* ({<identifier>}*(,)) = <expression> in <expression>

<primitive> ::= + | - | * | / | add1 | sub1 |zero? | list | cons | null | car | cdr | null?