CS315 Course Project I

Subhan Ibrahimli
Cavid Gayibli
Selim Can Gulsever

March 4, 2019

Introduction

The project is about the design of a new programming language for propositional calculus.This newly designed language will be similar to imperative languages. The main difference is that this new language will be specifically for propositional calculus. This part of the project is about the design of the language and the implementation of its lexical analyzer.

BNF

Backus-Naur Form (BNF) is a syntax for describing syntax. It's used to write a formal representation of a context-free grammar. In computer science, BackusNaur form or Backus normal form (BNF) is a notation technique for context-free grammars, often used to describe the syntax of languages used in computing, such as computer programming languages, document formats, instruction sets and communication protocols. In that part, there will be demonstrated our programming language BNF.

Description of Non-Terminals

In this section, there will be illustrated the descriptions of the given BNF forms.

Conclusions

In this section, we describe the results

1.BNF

1. <main_program> ::= LETSROLL LEFT_PARANT RIGHT_PARANT LEFT_BRACE NEW_LINE <program> RIGHT_BRACE

2. <program> ::= <statements> | <statement> <statements>

3. <statements> ::= <statement><new_line> | <statement><new_line><statements>

4. <statement> ::= <while> | <if> | <for> | <single_state> | <function_type>

5. <new_line> ::= NEW_LINE | NEW_LINE <new_line> | <comment>

6. <while> ::= WHILE LEFT_PARANT <expression> RIGHT_PARANT LEFT_BRACE <statements> RIGHT_BRACE

7. <if> ::= IF LEFT_PARANT <expression> RIGHT_PARANT LEFT_BRACE NEW_LINE <statements> RIGHT_PARANT | IF LEFT_PARANT <expression> RIGHT_PARANT LEFT_BRACE NEW_LINE <statements> RIGHT_BRACE ELSE LEFT_BRACE NEW_LINE <statements> RIGHT_BRACE

8. <for> ::= FOR LEFT_PARANT <declaration> COLON <expression> COLON <assignment_op> RIGHT_PARANT LEFT_BRACE NEW_LINE <statements> RIGHT_BRACE

9. <single_state> ::= <assign_state> | <declaration_state> | <return_state> | <function_state>

10. <assign_state> ::= <variable_name> ASSIGN_OP <expression>

11. <declaration_state> ::= <var_type> <assign_state> | <var_type> <var_names>

12. <expression> ::=

    <term> <low_op> <expression>

    | <term>

    | <term> <comp_op> <expression>

    | <term> <prop_op_med> <expression>

    | <term> <prop_op_low> <expression>

```
13. <term> ::=
        <var_name> <high_op> <term>
        | <prop_op_high> <variable_ident>
   | <integer> <high_op> <term>
   | <float> <high_op> <term>
   | <var_name>
   | <integer>
   | <float>
   | <string>
14. <low_op> ::= PLUS | MINUS
15. <high_op> ::= MULTIPLY | DIVISION | EXCLUSIVE_OR
16. <prop_op_high> ::= NEGATION
17. <prop_op_med> ::= DISJUNCTION | CONJUNCTION
18. <prop_op_low>  ::= IMPLICATION | DOUBLE_IMPLICATION
19. <comp_op> ::=
        ASSIGNMENT_OP
   | SMALL
   | GREAT
   | EQUALITY_CHECK
   | SMALL_OR_EQUAL
   | GREAT_OR_EQUAL
   | NOT_EQUAL
20. <var_name> ::= VAR_NAME
21. <var_names> ::= <var_name> | <var_name> COMMA <var_names>
22. <var_type> ::= TYPE_INT | TYPE_STRING | TYPE_FLOAT
23.  <function_state> ::=
        <var_name> LEFT_PARANT RIGHT_PARANT
        | <var_name> LEFT_PARANT <arguments> RIGHT_PARANT
24.  <return_state> ::= RETURN <expression> <new_line>
```

25. <arguments> ::= <argument> | <argument> COMMA <arguments>

26. <argument> ::= <integer> | <float> | <string> | <var_name>

27. <function_type> ::= <non_void_function> | <void_function>

28. <integer> ::= INT

29. <float> ::= FLOAT

30. <string> ::= STRING

31. <non_void_function> ::= <var_type> <var_name> LEFT_PARANT RIGHT_PARANT LEFT_BRACE NEW_LINE <statements> <return_state> RIGHT_BRACE | <var_type> <var_name> LEFT_PARANT <parameters> RIGHT_PARANT LEFT_BRACE NEW_LINE <statements> <return_state> LEFT_BRACE

32 <void_function> ::= VOID <var_name> LEFT_PARANT RIGHT_PARANT LEFT_BRACE NEW_LINE <statements> <return_state> RIGHT_BRACE | VOID <var_name> LEFT_PARANT <parameters> RIGHT_PARANT LEFT_BRACE NEW_LINE <statements> <return_state> LEFT_BRACE

33. <parameters> ::= <var_type> <var_name> | <var_type> <var_name> COMMA <parameters>

34. <comment> ::= COMMENT <new_line>

35. <print> ::= PRINT LEFT_PARANT <expression> RIGHT_PARANT

36. <println> ::= PRINT_LINE LEFT_PARANT <expression> RIGHT_PARANT