Țugulea Alexa

FAF - 182

**Lexer**

**L F P C**

Teacher :

Vdovicenco Alexandru

2020

**Grammar:**

Identifiers : string that the programmer chooses (name of variable, name of function)

Keywords : main, break, case, char, const, continue, default, double,

else, float, for, if, int, long, return, switch, void, while, function, true, false

Separators : , ; ( ) { }[]’’

Math operators : + - \* / =

Relational operators : == != < > <= >=

Logical operators : && || !

Numerical value : 0 1 2 3 4 5 6 7 8 9.

String : “letter\digit”

Comments : %

**Derivation :**

Terminals = {function, Parameters, Statement, Expression }

Nonterminals = {Identifiers, Keywords, Separators, MathOperators, LogicalOperators, Numerical Value, String, Comments}

Productions :

Grammar -> function End of file

function -> identifier “(”Parameters”)” Statement

Parameters ->Identifier”,” Parameters

Statement -> “{” Statement “S”

Statement -> “if” “(” Expression “)” Statement

Statement -> “else” Statement

Statement -> “else” “(” Expression “)” Statement

Statement -> “while” “(” Expression “)” Statement

Statement -> “return” Expression”;”

Expression -> String

Expression -> Numeric value

Expression ->Identifier

Expression ->”(“ Expression“)”

Expression ->”(“ Expression,“)”

Expression -> Expression ”[“ Expression“]” %indexing

Expression -> Expression ”(“ Expression,“)” %method call

Expression -> Expression “=” Expression

Expression -> Expression “+” Expression

Expression -> Expression “-” Expression

Expression -> Expression “+=” Expression

Expression -> Expression “-=” Expression

Expression -> Expression “||” Expression

Expression -> Expression “&&” Expression

Expression -> Expression “!=” Expression

Expression -> Expression “,” Expression

Expression -> “!” Expression

Expression -> “++” Expression

Expression -> “--” Expression

Expression -> Expression “++”

Expression -> Expression “--” Expression