G = (N, Σ, P, S)

S = program

N = { decllist, declaration, type, type1, listdecl, cmpdstmt, stmt, simplstmt, assignstmt, structstmt, whilestmt, ifstmt, condition, expr, term, factor, relation, operation }

Σ = { ":", ";", "(", ")", "read", "write", "int", "char", "str", "[", "]", "bool", "list", "=", "+", "*", "float", "finish", "else", "{", "}", "if", "loop", "START", "STOP", "<", "<", "<=", "<>", ">=", ">", "==", "!="

P={

program ::= "START" decllist "." cmpdstmt "END"

decllist ::= declaration | declaration decllist

declaration ::= IDENTIFIER "(" type ")" "."

type ::= type1|listdecl

type1 ::= "bool" | "char" | "str" | "float" | "int"

listdecl ::= "list" "(" type1 ")"

cmpdstmt ::= {stmt}

stmt ::= simplstmt | structstmt "."

simplstmt ::= iostmt | assignstmt | finish

iostmt ::= ("read" identifier) | ("write" identifier) "."

assignstmt ::= IDENTIFIER "=" expr "."

structstmt ::= cmpdstmt|ifstmt|whilestmt

whilestmt ::= "loop" ifstmt

ifstmt ::= "if" ":" condition "[" cmpdstmt "]" "."

condition ::= expr relation expr

expr ::= [expr("+"|"-")]term "."

term ::= term ("*"|"/") factor | factor "."

factor ::= "(" expr")" | IDENTIFIER | CONSTANT

relation ::= "<" | "<=" | "=" | "<>" | ">=" | ">"

}

Documentation:

Fisier = G , S, N, Σ , P

G – is the definition of the grammar (N, Σ, P, S)

S - is the starting symbol (the syntactical construct of the program)

N - is the non-terminal (declaration, statement, expression, term, factor,...)

Σ -is the terminals (identifiers, constants, operators, separators, reserved words)

P - is the syntactical rules (BNF style)