(Mini) Language Specification

Alphabet:

a. Upper (A-Z) and lower case letters (a-z) of the English alphabet
b. Underline character '_'
c. Decimal digits (0-9)
d. Special characters: * % ! # $ & | < > / ?

Lexic:
a. Special symbols, representing:
  - operators:
        arithmetic: +,-,*,%
        assignment: =
        relational: ==, <, <= ,>= ,> ,!=
  - separators: [] {} () , ; space
  - reserved words:
        if else while return void
        int char read write
        public static SimonaHalep
        stop

b. identifiers
  - a sequence of letters and digits, such that the first character is a letter and the length has at most 8 characters; the rule is:
        identifier = letter {letter | digit}
        letter = "A" | "B" |...| "Z" | "a" | "b" |...| "z"
        digit = "0" | "1"| "2"| ... | "9"

c. constants
  1. integer-rule:
        integer = "0" | ["+" | "-"] non_zero_digit {digit}
        non_zero_digit = "1" | "2"| ... | "9"
        digit = "0" | non_zero_digit
  2.character
        char = 'letter' | 'digit'
  3.string
        string = " " " {char_aux} " " "
        char_aux = letter | digit

Syntax:
  Syntactical rules:
    program = "public" "static" "void" "SimonaHalep" "(" ")" "{" statement_list "}"
    statement_list = statement | statement ";" statement_list
    statement = declaration | simple_statement | struct_statement | break_statement
    declaration = type identifier ";"
    type = simple_type | array_type
    simple_type = "int" | "char"
    array_type = simple_type "[" "]"
    simple_statement = assignment_statement | io_statement
    assignment_statement = IDENTIFIER "=" expression ";"
    expression = term | expression operation expression | "(" expression operation expression ")"
    term = IDENTIFIER | CONSTANT
    operation = "+" | "-" | "*" | "/" | "%"
    io_statement = input_statement | output_statement
    input_statement = "read" "(" IDENTIFIER ")" ";"
    output_statement = "write" "(" IDENTIFIER ")" ";" | "write" "(" CONSTANT ")" ";"

struct_statement = if_statement | while_statement
if_statement = “if” “(” condition “)” “{” statement_list “}” | “if” “(” condition “)” “{” statement_list “}” “else” “{” statement_list “}”
while_statement = “while” condition “{” statement_list “}”
condition = expression relation expression
relation = "<" | "<=" | "=" | "!=" | ">=" | “>”
break_statement = “break” “;”

Token type
identifier
constant
if
else
while
return
void
int
char
read
write
public
static
SimonaHalep
stop
+
-
*
/
%
=
==
!=
!
<
<=
>=
>
[
]

{
}
(
)
,
;
$