Mini-Language Specification

Alphabet

- A-Z, a-z (Uppercase and lowercase letters)
- 0-9 (Digits)
- _

Operators

- +, -, *, /, % (Addition, Subtraction, Multiplication, Division, … - Arithmetic)
- ==, !=, >, < (Equality, Inequality, Greater, Equal - Relational)
- &&, ||, ! (And, Or, Not - Logical)
- = (Assignment)

Separators (for this document's readability each separator will be in "")

- ";", ",", " ", "{", "}", "(", ")"

Keywords

- read
- write
- if
- else
- for
- while
- break
- integer
- string
- character
- array
- return

Identifiers

- identifier = letter {letter|digit} | letter
- letter = A| B| …| Z| a| b| …|z
- digit = 0| non_zero_digit
- non_zero_digit = 1| … 9

Constants

- integer = 0 | non_zero_digit {digit}
- character = 'letter'|'digit'
- string = "{letter|digit}"

Token	code
Identifier	0
Constant	1
[	2
]	3
{	4
}	5
;	6
:	7
,	9
<	10
>	11
==	30
!=	31
!	32
&&	33
||	34
=	35
+	36
-	37
*	38
/	39
%	40
character	60
integer	61
string	62
array	63
if	64
else	65
for	66
while	67
break	68
return	69
read	70
write	71

Syntax

- declaration = type " " identifier
- simple_type = "integer" | "string" | "character"
- array_declaration = simple_type " " "array" "[" integer "]"
- type = simple_type | array_declaration
- compound_statement = "{" statement_list "}"
- statement_list = statement| statement ";" statement_list
- statement = simple_statement | struct_statement
- simple_statement = assign_statement | io_statement | declaration
- struct_statement = compound_statement | if_statement | while_statement | for_statement
- if_statement = "if" condition statement ["else" statement]
- for_statement = "for" "(" "number" assign_statement ";" condition ";" assign_statement ")" statement
- while_statement = "while" condition statement
- assign_statement = Identifier "=" expression
- expression = [expression ("+"|"-")] term
- term = term ("*" | "/") factor | factor
- factor = "(" expression ")" | integer | Identifier| Identifier "[" integer "]"
- io_statement = ("read" IDENTIFIER) | ("write" (Identifier| Constant))
- condition = "(" expression relation expression ")"
- relation = "<" | "<=" | "==" | "!=" | ">=" | ">"