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);

Lexic:
a.Special symbols, representing

-operators : + - * % ^ / = < <= == != >= > and or not
-separators : [] : ; , space
-reserved words: number boolean character string if then else for read write do define while

b.identifiers
- a sequence of letters and digits, such that the first character is a letter or "_"; the rule is:
identifier = ["_"]letter[{letter | digit}]
letter = "A" | "B" | . ..| "Z" | "a" | "b" | ... | "z"
digit = "0" | "1" | ... | "9"

c.constants

1.number:
number = ["+"|"-"]nonzero{digit} | 0
digit = "0" | nonzero
nonzero = "1" | "2" | ... | "9"

2. character
character = 'letter'| 'digit'

3.string
string='char{string}'
character = 'letter'| 'digit'

Syntax:
The words - predefined tokens are specified between " and ":

program = "~"declist "~" cmpdstmt
declist = declaration | declaration "\n" declist
declaration = "define" type IDENTIFIER
type = "BOOLEAN" | "CHARACTER" | "NUMBER" | "STRING"

cmpdstmt = "start" stmtlist "end"
stmtlist = stmt | stmt "\n" stmtlist
stmt = simplestmt | structstmt

simplestmt = assignstmt | iostmt
assignstmt = IDENTIFIER "=" expression
expression = expression "+" term | term
term = term "*" factor | factor
iostmt = "READ" | "WRITE" "("IDENTIFIER")"

structstmt = cmpdstmt | ifstmt | whilestmt
ifstmt = "IF" condition "DO:" stmt ["ELSE:" stmt]
whilestmt = "WHILE" condition "DO:" stmt
condition = expression RELATION expression
RELATION = "<" | "<=" | "==" | "!=" | "=>" | ">"

+
-
*
%
^
/
=
<
<=
==
!=
>=
>
:
;
and
or
not
number
boolean
character
string
if
then
else
for
read
write
do
define
while