Lexic.txt

Alfaphabet:

       a). lowercase letters from the English alphabet (a-z)

       b). decimal digits (0-9)

Lexic:

 1. Special symbols, representing:

       a). separators: [], {}, /, :, space,

       b). reserved words: sir => array,

               intreg => int,

               real => float,

               dublu => double,

               lung => long

               caracter => char,

               maimare => >,

               maimic => <,

               inca un pic => while

               pt => for,

               daca => if,

               dacanu => else,

               dainapoi => return,

               sa vad => print,

               lungime => length()

       c). operators: ##, -, :, =:=, =!:=, ",",

2. Identifiers:

      a). seq of letters and digits stat the first character is a letter ->

identifier ::= letter | letter{letter}{digit}

letter ::= "A" | "B" | . ..| "Z" | "a" | "b" | . ..| "z"

digit ::= "0" | "1" |...| "9"

      b). constants ~ integer -> nrconst:= "0" | ["+" | "-"]

                         nonzerodigit { "0" | nonzerodigit}

                         nonzero_digit = "1" |...| "9"

          ~ characters -> character:="letter"|"digit"

          ~ string -> constchar:="string"

                  string:=char{string}

                  char:=letter|digit

Syntax.in:

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

program = statementlist

statementList = statement ";" statement

statement= dec | simplestatement | structstatement

simplestatement = assignmentStatement | outStatement

assignmentStatement = identifier "=:=" expression

outStatement = ("read" |"savad") "(" identifier ")"

structStatement = ifStatement | whileStatement | forStatement | cmpStatement

ifStatement = daca" "[" condition "]" stmt {dacanu stmt}

condition = expression RELATION expression

expression = term | expression operation expression | "(" expression operation expression ")"

operation = "aduna" | "-" | "x" | ":" | "&"

term = identifier | nr

whileStatement = "incaunpic" condition compstmt

forStatement = "pt" "[" assignment; condition; "]" stmt

cmpStatement = "{" statementList "}"

dec = type identifier

type = mainType | arrayType

mainType = "intreg" | "lung" | "real" | "dublu" | "caracter"

arrayType = mainType "[" elements "]"

tokens.in:

0 identifier

1 constants

2 "["

3 "]"

4 "{"

5 "}"

6 "/"

7 ":"

8 " "

9 "##"

10 "-"

11 "=:="

12 "=!:="

13 ","

14 "sir"

15 "intreg"

16 "real"

17 "dublu"

18 "caracter

19 "mai mare"

20 "mai mic"

21 "inca un pic"

22 "pt"

23 "daca"

24 "dacanu"

25 "dainapoi"

26 "sa vad"

27 "lungime"