Language specification:

1. Language Definition:
   1. Alphabet

1.1.a Upper (A-Z) and lower case (a-z) letters of the English alphabet;

b Underline character ‘\_’;

c Decimal digits (0-9);

Lexic:

1.Special symbols, representing:

-operators: + - / \* = := <= >= < >

-separators [ ] { } : ; space

-reserved words: arr(array) chr(char) integer(int) @(const) whether(if) next(then) make(do) else of program in(read) out(write) name(var) while

2.Identifiers

-a sequence of letters and digits, such that the first character is a letter;

The rule is:

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

letter ::= “a” | “b” | … | “z” | “A” | “B” | … | “Z”

digit ::= “0” | “1” | … | “9”

3.Constants

a. Integer:

noconst ::= +no | -no | no

no ::= digit | non\_zero\_digit{digit}

non\_zero\_digit ::= “1” | “2” | … | “9”

2.Character:

character ::= ‘letter’|‘digit’

3.String:

constchar ::= “string”

string ::= char{string}

char::= letter | digit

2.2 Syntax:

The words – predefined tokens are specified between “ and “:

1. Sintactical rules:

program ::= “NAME” decllist “;” cmpdstmt “.”

decllist ::= declaration | declaration “;” decllist

declaration ::= IDENTIFIER “:” type

type1 ::= “BOOLEAN” | “CHAR” | “INTEGER” | “REAL”

arraydecl ::= “ARR” “[“nr”]” “OF” type1

type ::= type1|arraydecl

cmpdstmt ::= “BEGIN” stmtlist “END”

stmtlist ::= stmt | stmt “;”stmtlist

stmt::= simplstmt | structstmt

simplestmt ::= assignstmt | iostmt

assignstmt ::= IDENTIFIER “:=” expression

expression ::= expression “+” term | expression “-“ term | term

term ::= term “\*” factor | term “/” factor | factor

factor ::=”(“expression”)” | IDENTIFIER

iostmt::= “IN” | “OUT” “(“IDENTIFIER”)”

structstmt::=cmpdstmt | ifstmt | whilestmt

ifstmt::= “WHETHER” conditition “NEXT” stmt [“ELSE” stmt]

whilestmt::= “WHILE” condition “MAKE” stmt

condition::=expression RELATION expression

relation::=”<” | “>” | “=” | “<=” | “>=” | “<>”