Specificare minilimbaj de programare – subset FreeBASIC

1. Definirea limbajului

1.1. Alfabet

a. Literele mari si mici ale alfabetului englez (A-Z, a-z);

b. Caracterele ’_’,’+’,’-’,’*’,’/’,’&’,’=’,’<’,’>’,’(’,’)’,’;’,’:’,’$’,’ ‘ ‘, ‘^’,’&’;

c. Cifrele zecimale (0-9).

1.2. Lexic

a. Simboluri speciale:

* operatori:

aritmetici: +, -, *, /, ^, MOD

de atribuire: =, +=, -=, *=, /=, ^=, =&

de egalitate: =, <>

de ordine: <, >, <=, >=

logici: AND, OR, XOR, NOT

de referentiere: $

de concatenare: &

* separatori: (,), ;, :, ,, spatiu, ‘

b. Cuvinte rezervate:

DIM, CONST, AS, INTEGER, STRING, BOOLEAN, DOUBLE, LONG, PERECHE, STR, IF, THEN, END IF, END, FOR, TO, NEXT, PRINT, INPUT, GOTO, TYPE, END TYPE, SWAP, WHILE, WEND

c. Identificatori:

identificator ::= litera[6{cifra|litera|’_’}(cifra|litera)]

litera ::= ”a” | ”b” | ... | ”z” | ”A” | ”B” | ... | ”Z”

cifra ::= ”0” | ”1” | ... | ”9”

d. Constante:

1.intregi:

const_int ::= [semn]cifra_nenula{cifra}|”0”

semn ::= ”+”|”-”

cifra_nenula ::= ”1”|...|”9”

cifra ::= “0” | cifra_nenula

2.string:

const_str ::= ” ’ ”caracter{caracter}” ’ ”
caracter ::= litera|cifra|”_”|” ”

3.reale:

const_re ::= (const_int|const_int”.”{“0”}constnat) |
[semn]”0””.”{”0”}[const_nat]cifra_nenula
const_nat ::= cifra_nenula{cifra}|”0”
const_hex ::= “&””H”cifra_hexa{cifra_hexa}
cifra_hexa ::= cifra | “A” | “B” | ... | “F”

2. Sintaxa

2.1 Reguli sintactice

program ::= numar_linie descriere {comanda} end

descriere ::= comentariu

comanda ::= numar_linie [eticheta”:”] instructiuni

eticheta ::= litera[{litera|’_’}]{litera}

numar_linie ::= cifra_nenula cifra

instructiuni ::= instructiune {“:” instructiune}

instructiune ::= declarare | atribuire | conditionala | for | afisare | citire | salt | comentariu | definire_pereche | swap | “END” | **while**

declarare ::= “DIM” | “CONST” identificator “AS” tip [“=” expresie]

tip ::= “INTEGER” | “STRING” | “BOOLEAN” | “DOUBLE” | “LONG” | “PERECHE”

atribuire ::= identificator [operator]”=” expresie

expresie ::= constanta | identificator | expresie operator expresie | conversie

conversie ::= (“STR”“(“constanta”)”) | (“STR””$””(“identificator”)”)

conditionala ::= “IF” conditie “THEN” instructiuni “END IF” conditie ::= (expresie relatie expresie) | (conditie operator_binar conditie) | (operator_unar conditie)

for ::= “FOR” identificator [“AS” “INTEGER”|”LONG”] “=” const_int “TO” const_int {instructiuni} “NEXT” identificator

while ::= “WHILE” conditie instructiuni “WEND”

afisare ::= “PRINT” expresie {“;” expresie}

citire ::= “INPUT” const_string”,” identificator

salt ::= “GOTO” numar_linie | eticheta

comentariu ::= “ ’ ” text

definire_pereche ::= “TYPE” “PERECHE” “stanga” “AS” “INTEGER” “dreapta” “AS” “INTEGER” “END TYPE”

swap ::= “SWAP” identificator ”,” identificator

end ::= numar_linie “END”

2.2 Reguli lexicale

identificator ::= litera[{cifra|litera|’_’}][{cifra|litera}]

constanta ::= const_int | const_re | const_nat | const_str | const_hex

const_int ::= [semn]cifra_nenula{cifra}|”0”

const_re ::= (const_int|const_int”.”{”0”}const_nat) |[semn]”0””.”{”0”[const_nat]cifra_nenula

const_nat ::= cifra_nenula{cifra}|”0”

const_hex ::= “&””H”cifra_hexa{cifra_hexa}

cifra_hexa ::= cifra | “A” | “B” | ... | “F”

const_str ::= ” “ ” text ” “ ” [“&” const_str | conversie]

text ::= caracter{caracter}

caracter ::= litera|cifra|”_”|” ”

litera ::= ”a” | ”b” | ... | ”z” | ”A” | ”B” | ... | ”Z”

cifra ::= “0” | cifra_nenula

cifra_nenula ::= “1” | … | “9”

operator ::= “+” | “-” | “*” | “/” | “MOD” | “^”

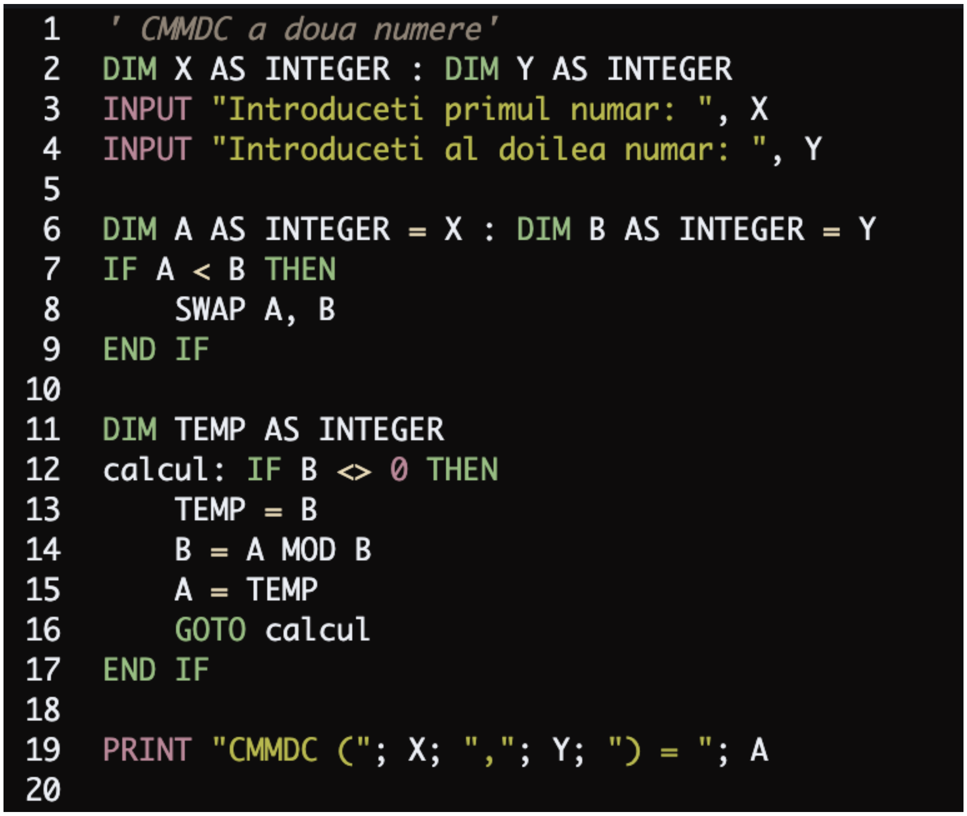
operator_binar ::= “AND” | “OR” | “XOR”

operator_unar ::= “NOT”

relatie ::= “<” | “>” | “<=” | “>=” | “<>” | “=”

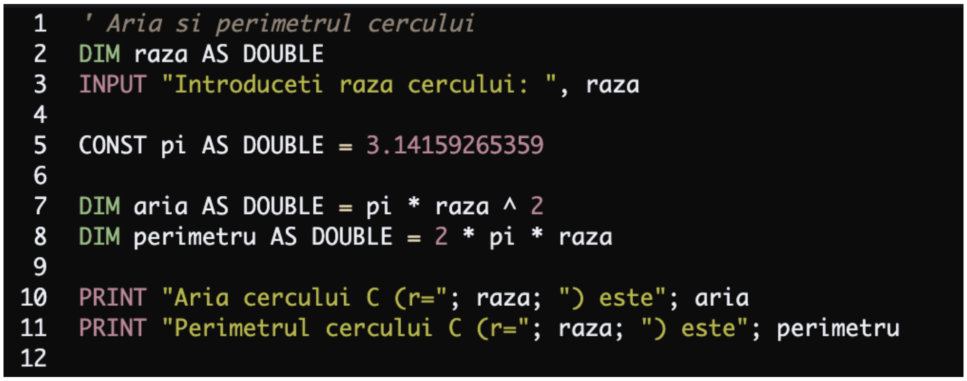
3. Exemple de programe simple

CMMDC-ul a 2 numere naturale



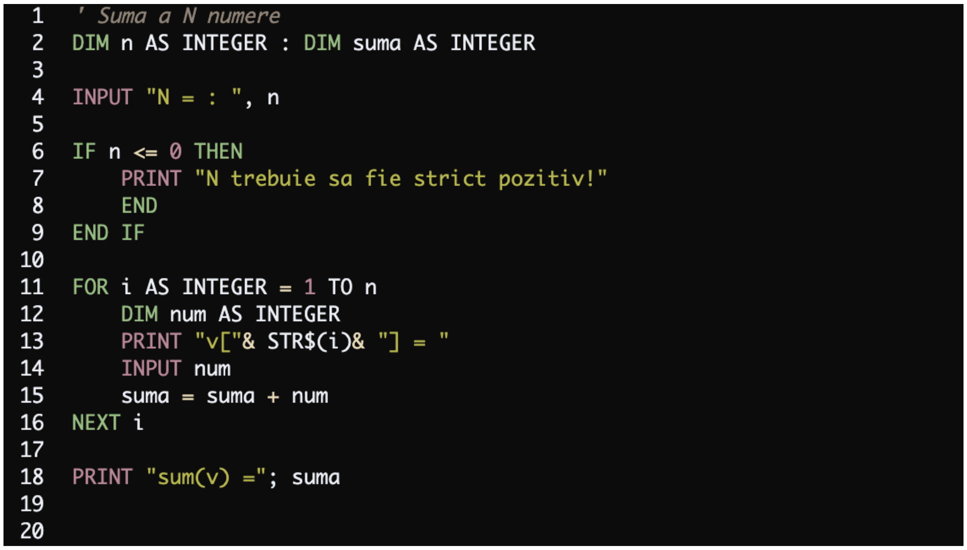
```
' CMMDC a doua numere'
DIM X AS INTEGER : DIM Y AS INTEGER
INPUT "Introduceti primul numar: ", X
INPUT "Introduceti al doilea numar: ", Y

DIM A AS INTEGER = X : DIM B AS INTEGER = Y
IF A < B THEN
    SWAP A, B
END IF

DIM TEMP AS INTEGER
calcul: IF B <> 0 THEN
    TEMP = B
    B = A MOD B
    A = TEMP
    GOTO calcul
END IF

PRINT "CMMDC ("; X; ","; Y; ") = "; A

```

Perimetrul si aria cercului



```
' Aria si perimetrul cercului
DIM raza AS DOUBLE
INPUT "Introduceti raza cercului: ", raza

CONST pi AS DOUBLE = 3.14159265359

DIM aria AS DOUBLE = pi * raza ^ 2
DIM perimetru AS DOUBLE = 2 * pi * raza

PRINT "Aria cercului C (r="; raza; ") este"; aria
PRINT "Perimetrul cercului C (r="; raza; ") este"; perimetru

```

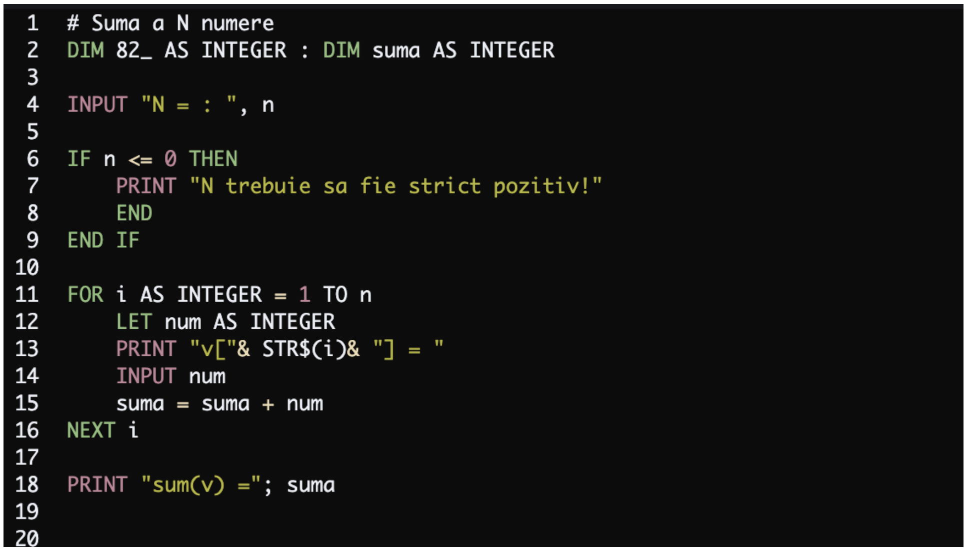
Suma a n numere citite de la tastatura



```
' Suma a N numere
DIM n AS INTEGER : DIM suma AS INTEGER

INPUT "N = : ", n

IF n <= 0 THEN
    PRINT "N trebuie sa fie strict pozitiv!"
    END
END IF

FOR i AS INTEGER = 1 TO n
    DIM num AS INTEGER
    PRINT "v["& STR$(i)& "] = "
    INPUT num
    suma = suma + num
NEXT i

PRINT "sum(v) ="; suma
```

4. Exemple de programe cu erori

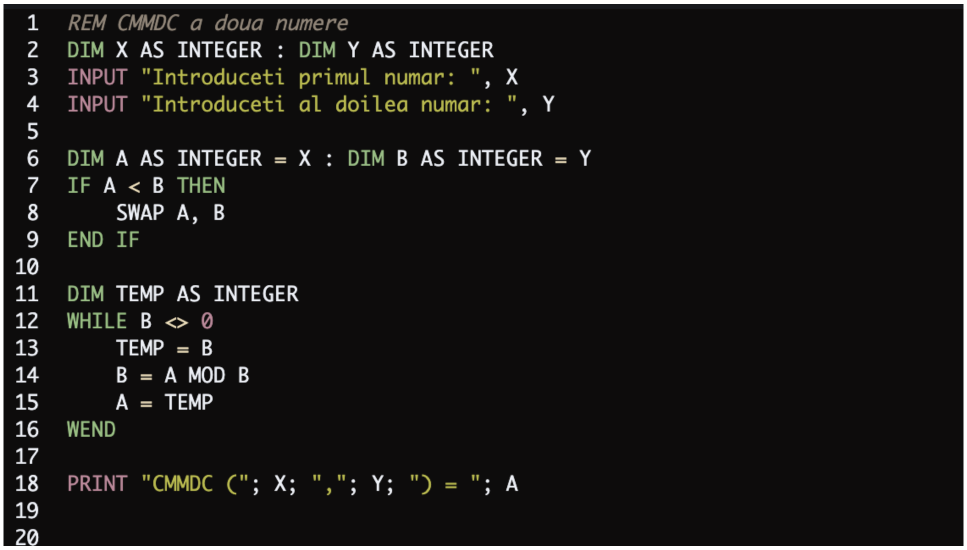
a. Erori FreeBASIC



```
# Suma a N numere
DIM 82_ AS INTEGER : DIM suma AS INTEGER

INPUT "N = : ", n

IF n <= 0 THEN
    PRINT "N trebuie sa fie strict pozitiv!"
    END
END IF

FOR i AS INTEGER = 1 TO n
    LET num AS INTEGER
    PRINT "v["& STR$(i)& "] = "
    INPUT num
    suma = suma + num
NEXT i

PRINT "sum(v) ="; suma
```

82_ nu este un nume valid de identificator, LET nu este definit

b. Erori MLP



```
REM CMMDC a doua numere
DIM X AS INTEGER : DIM Y AS INTEGER
INPUT "Introduceti primul numar: ", X
INPUT "Introduceti al doilea numar: ", Y

DIM A AS INTEGER = X : DIM B AS INTEGER = Y
IF A < B THEN
    SWAP A, B
END IF

DIM TEMP AS INTEGER
WHILE B <> 0
    TEMP = B
    B = A MOD B
    A = TEMP
WEND

PRINT "CMMDC ("; X; ","; Y; ") = "; A
```

REM nu este definit, WHILE nu este definit