LABORATOR 1 – LFTC

I:Text sursa pentru programe date

**1.Calculeaza perimetrul si aria cercului de o raza data**

Function arie

begin

Var a:real;

Var p:real;

Var r:real;

Const pi:real;

pi=3.14;

read(r);

a=pi\*r\*r;

p=2\*pi\*r;

write(a);

write(p);

end

**2.Determina CMMDC a doua numere naturale**

Function cmmdc

Var a:integer;

Var b:integer;

begin

if b==0{

b=a;

write(b);

}

else{

call:cmmdc(b:integer a%b:integer);

}

end

**3.Calculeaza suma a n numere reala citite de ela tastatura**

Function suma

Var n:integer;

begin

var suma:integer;

suma=0;

var numar:integer;

for: var i=1,n,1{

read(numar);

suma = suma + numar;

}

Write(suma);

end

II:Specificarea minilimbajului de programare

**1.Alfabetul limbajului**

a)Litere mari ale alfabetului englez(A-Z)

b)Litere mici ale alfabetului englez(a-z)

c)Cifrele sistemului zecimal(0-9)

**2.Lexic**

a)Simboluri pentru reprezentarea:

* Operatorilor: +, -, /, \*, %, <, >, <=, >=, !=, ==, =
* Separatorilor: ;, (, ), {, }, blank, :, ,, [, ]
* Cuvinte rezervate: read,write,function,if,else,begin,end,for,while,real,integer,string,array,OR,AND, var,const,call

b)Identificatori.Reguli lexicale

* Identificator:litera|litera{litera}{cifra}
* Cifra:”0”|”1”|”2”|”3”|”4”|”5”|”6”|”7”|”8”|”9”
* Relatie: „<”|”<=”|”!=”|”>”|”>=”|”==”
* Integer: „0”[„+”|”-„]cifraz{cifra}
* Cifraz: ”1”|”2”|”3”|”4”|”5”|”6”|”7”|”8”|”9”
* Real:[„+”|”-„] {cifra}”.”{cifra}cifraz|{cifra}
* Const:real|integer|string

**3.Sintaxa limbajului**

a)reguli statice

* Function = „function” identificator declarParam „begin” declarParam listaInstrutiuni „end”
* declarParam={declarare}
* declarare = [„var”|”const”]identificator „:”type”;”
* type:type1|type2
  + - type1:”integer”|”real”|”string”
    - type2:”array[{„constanta”}] „of” type1
* listaInstrutiuni = {instructiune}
  + instructiune = atribuire|io|if|while|for|call
    - atribuire = identificator”=”expresie”;”
      * expresie=expresie(„+”|”-„)term|term
        + term=term(„\*”|”/”|”%”)factor|factor

factor=”(„expresie”)”identificator| constanta |identifcompus

identifcompus=identificator „[„constanta”]”

* + - io=(„read”|”write”)”(„(identificator|identifcompus”)”)”
    - if=”if” conditie „{„listaInstructiuni „}”[„else””{„listaInstructiuni „}”]
      * conditie=expresie relatie expresie
    - while=”while”conditie”{„ listaInstructiuni „}”
    - for=”for” identificator „=”constanta|identificator”,”constanta| identificator”,”[constanta|identificator]”{„listaInstructiuni”}”
    - call=”call””:” identificator”(„{[identificator|expresie] „:” type}”)”

**4.Codificarea**

|  |  |
| --- | --- |
| Tip Atom | Cod |
| identificator | 0 |
| constanta | 1 |
| function | 2 |
| const | 3 |
| var | 4 |
| begin | 5 |
| end | 6 |
| if | 7 |
| else | 8 |
| for | 9 |
| while | 10 |
| integer | 11 |
| real | 12 |
| array | 13 |
| of | 14 |
| read | 15 |
| write | 16 |
| call | 17 |
| ; | 18 |
| ( | 19 |
| ) | 20 |
| { | 21 |
| } | 22 |
| , | 23 |
| + | 24 |
| - | 25 |
| \* | 26 |
| / | 27 |
| % | 28 |
| > | 29 |
| < | 30 |
| >= | 31 |
| <= | 32 |
| = | 33 |
| != | 34 |
| == | 35 |
| [ | 36 |
| ] | 37 |
| . | 38 |
| : | 39 |
| OR | 40 |
| AND | 41 |