**Laboratory 1 FLTC**

Alphabet:

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

b. Decimal digits (0-9);

Lexic:

a.Special symbols, representing:

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

- separators [ ] { } : ; space

- reserved words:

array char then else if int of program read while write Begin End char

**I)Sintactical rules**

<program> ::="Begin"[<list\_decl>]<list\_instr>"End"

<list\_decl> ::= <type> “ ”ID”;” |<type>” “<ID> ”;” <list\_decl> |<type> “ “<ID> “=” <expr>”;”

<type> ::=<simple\_type> | array “<” <simple\_type> “>” ”(“<term> “)”

<simple\_type> ::= "int" | "char"

<list\_instr> ::= <instr>|<instr><list\_intr>

<instr>= <instr\_io> |<instr\_attr> |<instr\_cond> |<instr\_cicl>

<instr\_io> ::= read “(“ ID “)” |write"(“ ID”)”

<instr\_attr> ::= <ID> “=” <expr>

<instr\_cond> ::= if “(“ <expr\_cond> “)” then “{“ <list\_instr> “}” |

if “(“ <expr\_cond> “)” then “{“ <list\_instr> “}”else “{“ <list\_instr> “}” |

if “(“ <expr\_cond> “)” then “{“ <list\_instr> “}”else <instr\_cond>

<expr\_cond> ::=<expr> <op\_cond> <expr>

<expr> ::= <expr> <op> “(“<expr>”)” | <term>

<term> ::= <ID> | <const>

<instr\_cicl> ::= while “(“ <expr\_cond> “)” “{“ <list\_instr> “}”

<const>::=<integer>|<character>

<digit>::="0"|<nonzero>

<integer>::=["+"|"-"]<nonzero>{digit}|"0"

**II) Lexical rules**

<ID> ::= [A-Za-z][A-Za-z0-9]\*

<nonzero>::="1"|..|"9"

<op> ::= “+” | “/” | “%” | “\*”|”-”

<op\_cond> ::= “==” | “>” | “<” | “!=” | “<=” | “>=”

<character>::=" ' "[A-Za-z0-9] " ' "

**III)Some examples**

**Perimeter and aria of a circle with known radius r**

Begin

Float raza,;

Float perimetrul;

Float aria;

Read(raza)

Perimetrul = 2\*(3,14\*(raza))

Aria = 3,14\*(raza\*(raza))

Write(perimetrul)

Write(aria)

End

**Greatest common diviser of 2 numbers**

Begin

Int a;

int b;

Int r;

Read(a)

Read(b)

R=a%b

While(r!=0)

{

R=a%b

A=b

B=r

}

Write(a)

End

**Sum of n integers**

Begin

Int n;

int x;

Int i;

Int sum;

Read(n)

I = 0

sum=0

While(i<n){

Read(x)

sum = sum + x

I= i+1

}

Write(sum)

End