1. **Definire MLP**

Voi alege C++ ca limbaj principal pentru MLP

\* 1. Tipuri de date:

-> integer: tip de date reprezentand numere intregi

-> float: tip de date reprezentand numere reale

-> string: tip de date reprezentand siruri de caractere

\* 2. Instructiuni:

-> instructiune de atribuire: <-

-> intrsuctiune de intrare / iesire: write(expresie) / read(variabila)

-> instructiune de selectie: if(conditie) { } [else { }]

-> instructiune de ciclare: while(conditie) { }

\* 3. Restrictii:

-> Identificatorii trebuie sa fie unici si diferiti de cuvintele cheie. Pot contine caractere alfanumerice si '\_', dar nu pot incepe cu '\_'.

-> Constantelor nu li se pot atribui valori.

\* 4. Analiza lexicala:

-> alfabet: litere si majuscule din alfabetul englez (a-z), cifre arabe (0-9) si caracterul '\_'.

-> simboluri: -aritmetice: +, -, \*, /, %

-asignare: <-

-egalitate: ==, !=

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

-de secventa: ,

-> separatori: () {} ; :

-> cuvinte rezervate: -int = int

-float = float

-string = string

-read = cin>> / scanf()

-write = cout<< / printf()

-if = if

-else = else

-while = while

-const = const

-nothing = void

\* 5. Identificatori:

-> identificator = alfanumeric [alfanumeric]\*

-> alfanumeric = litera | cifra

-> litera = "a" | "b" | ... | "z" | "A" | "B" | ... | "Z"

-> cifra = "0" | "1" | ... | "9"

-> semn = ["+" | "-"]

-> virgula = ","

\* 6. Constante:

-> int = [semn] cifra {cifra}

-> string = '{litera | cifra}'

-> float = [semn] cifra {cifra} [virgula cifra {cifra}]

\* 7. Tabela de simboluri:

ATOM | COD

----------------------

ID | 0

CONST | 1

( | 2

) | 3

{ | 4

} | 5

, | 6

; | 7

int | 8

float | 9

string | 10

+ | 11

- | 12

\* | 13

/ | 14

% | 15

<- | 16

== | 17

!= | 18

< | 19

> | 20

<= | 21

>= | 22

: | 23

write | 24

read | 25

if | 26

else | 27

while | 28

const | 29

nothing | 30

' | 31

" | 32

main | 33

\* 8. Definere BNE

<program> ::= main() <corp>

<functie> ::= <antet\_functie> <corp>

<antet\_functie> ::= <tip> IDENTIFICATOR (<parametrii>)

<tip> ::= int | float | string

<parametrii> ::= [{<tip> IDENTIFICATOR}\*]

<declarare> ::= <tip> IDENTIFICATOR | <tip> <atribuire>

<atribuire> ::= IDENTIFICATOR <- <expresie>

<expresie> ::= IDENTIFICATOR | CONSTANTA | <expresie> {<operator\_aritmetic> | <operator\_relational> <expresie>

<operator\_aritmetic> ::= + | - | / | % | \*

<operator\_relational> ::= > | < | >= | <= | == | !=

<corp> ::= { <instructiune> }\*

<instructiune> ::= <declarare> | <instructiune\_citire> | <instructiune\_scriere> | <instructiune\_if> | <instructiune\_while>

<instructiune\_citire> ::= read(ID)

<instructiune\_scriere> ::= write(ID)

<instructiune\_if> ::= if(<expresie>) <corp> [else <corp>]

<instructiune\_while> ::= while(<expresie>) <corp>

IDENTIFICATOR ::= ^(a-zA-Z0-9)[\_a-zA-Z0-9]{0,29}

CONSTANTA ::= <constanta\_int> | <constanta\_float> | <constanta\_string>

<constanta\_int> ::= ^[+-]?[0-9]+$

<constanta\_float> ::= ^[+-]?[0-9]+(\.[0-9]+)?$

<constanta\_string> ::= ^'.\*'$

**II.** **Miniprograme**

1.main()

{

float r;

float PI = 3,14;

read(r);

float perimetru = 2 \* PI \* r;

float arie = PI \* r \* r;

string rezultat = perimetru + " " + arie;

write(rezultat);

}

2.

main()

{

int a, b;

read(a);

read(b);

if (b == 0)

{

write(0);

}

else {

int rest = a % b;

while (rest != 0)

{

a = b;

b = rest;

rest = a % b;

}

write(b);

}

}

3.

main()

{

int suma = 0;

int n;

read(n);

while (n >= 0)

{

int nr;

read(nr);

suma = suma + nr;

n = n - 1;

}

write(suma);

}

**III. Erori**

1.

main()

{

aaa suma; -> eroare, tipul “aaa” nu exista

0 = suma; -> eroare nu se poate atribui valori constantelor

}

2.

main()

{

int i;

for (int i = 1; i < 9; i++) -> instructiunea “for” nu este definita

{

return 0; -> return nu este definit

}

}