

Lab 1 - LFTC - Abramiuc Andrei

1. Specificare MLP - bazat pe C++

1.1 Elemente lexicale

- **ID** (identificator): incepe cu litera mica `a..z` (restrictie MLP) urmat de `letter | digit`.
 - Regex MLP: `[a-z][A-Za-z0-9]*`
- **CONST_INT**:
 - Regex: `[+-]?[0-9]+`
- **CONST_REAL**: parte zecimala obligatorie, eventual semnata:
 - Regex: `[+-]?[0-9]+\.[0-9]+`
- **CONST_STRING**: intre ghilimele duble: `"[^"\n]*"`
- **Cuvinte rezervate (keyword)**: `int, float, string, if, else, while, cin, cout, return`
- **Operatori**: `=, +, -, *, /, %, ==, !=, <, >, <=, >=, <<, >>`
- **Separatori**: `, () { }`
- Spatii/tab/ newline = separatori (ignorate de lexer dar folosite pentru contorizare linii)

Daca `+` sau `-` apare **lipit** (fara spatiu) inaintea unei cifre, el **face parte din literal** si este parte a tokenului **CONST_INT** sau **CONST_REAL**. Daca exista **spatiu** intre semn si cifra, semnul este token **MINUS** si **5** este **CONST_INT**.

1.2 BNF

- `<program> ::= <decl-list> <stmt-list>`
- `<decl-list> ::= <decl> | <decl> <decl-list>`
- `<decl> ::= <type> <id-list> ;`
- `<type> ::= int | float | string`
- `<id-list> ::= ID | ID , <id-list>`
-
- `<stmt-list> ::= <stmt> | <stmt> <stmt-list>`
- `<stmt> ::= <assign-stmt> ;`
- `| <io-stmt> ;`
- `| <if-stmt>`
- `| <while-stmt>`
- `| { <stmt-list> }`
-
- `<assign-stmt> ::= ID = <expr>`
-
- `<io-stmt> ::= cin >> <input-list>`

- | cout << <output-list>
- <input-list> ::= ID | ID >> <input-list>
- <output-list> ::= <expr> | <expr> << <output-list>
-
- <if-stmt> ::= if (<cond>) <stmt> [else <stmt>]
- <while-stmt> ::= while (<cond>) <stmt>
-
- <cond> ::= <expr> <relop> <expr>
- <relop> ::= == | != | < | > | <= | >=
-
- <expr> ::= <term> { (+|-) <term> }
- <term> ::= <factor> { (* | /) <factor> }
- | <int-factor> % <int-factor>
-
- <factor> ::= CONST_INT | CONST_REAL | CONST_STRING | ID | (<expr>)
- <int-factor> ::= ID | CONST_INT | (<expr>)

2. Codurile sursa a 3 sub-programe

2.1 Perimetru si aria cercului de o raza data data

```
#include <iostream>
using namespace std;

int main() {
    float r;
    float pi;
    float perim;
    float area;

    cin >> r;

    pi = 3.141;
    perim = 2.0 * pi * r;
    area = pi * r * r;

    cout << "Perimetru = " << perim << "\n";
    cout << "Arie = " << area << "\n";
    return 0;
}
```

2.2 Cmmdc a 2 nr naturale

```
#include <iostream>

using namespace std;

inty main() {
    int a;
    int b;
    int r;

    cin >> a >> b;

    while (b != 0) {
        r = a % b;
        a = b;
        b = r;
    }
    cout << a << "\n";
    return 0;
}
```

2.3 Suma a n numere citite de la tastatura

```
#include <iostream>

using namespace std;

inty main() {
    int n;
    int i;
    float x;
    float s;

    cin >> n;

    i = 0;
    s = 0;

    while (i < n) {
        cin >> x;
        s = s + x;
        i = i + 1;
    }
}
```

```

    cout << s << "\n";
    return 0;
}

```

3. Textele sursa a doua programe care contin erori conform MLP-ului definit

3.1 Unul dintre programe contine doua erori care sunt in acelasi timp erori in limbajul original (C++)

```

#include <iostream>
using namespace std;

int main() {
    int a, b      // lipsa ;
    cin >> a >> b;
    c = a + b;    // 'c' nedeclarat
    cout << c << "\n";
    return 0;
}

```

3.2 Al doilea program contine doua erori conform MLP, dar care nu sunt erori in limbajul original

```

#include <iostream>
using namespace std;

int main() {
    int _count;      // valid in C++, INVALID in MLP (underscore)
    float Radius;   // valid in C++, INVALID in MLP (majuscula)
    cin >> _count >> Radius;
    float suma = _count * Radius;
    cout << suma << "\n";
    return 0;
}

```

4. Activitate laborator

<conditie> ::= if <cond> then <instr> else <instr>

<ciclare> ::= while <cond> do <instr>

<cond> ::= <expr> <op_rel> <expr>

<op_rel> ::= = | <> | < | <= | > | >=

,dar trebuie modificat la BNR-ul curent:

<instr> ::= <atribuire>

| <conditie>
| <ciclare>

```
var a, b: integer;
begin
  a := 30;
  b := 18;
  while a <> b do
    begin
      if a > b then
        a := a - b
      else
        b := b - a
    end
  end.
```