## 1. Specificarea mini limbajului de programare (EBNF)

program = header , declarations , '*begin'* , instruction_list , '*end*' ;

header = '*program*' , program_name , '*;*' ;

program_name = ID;

declarations = ['*type*' , user_defined_type , '*:*' , other_type , '*;*'] '*var*' , variable_list , '*:*' , data_type , '*;*' ;

variable_list = ID , { ',' , ID } ;

data_type = '*integer*' | '*real*' | user_defined_type ;

user_defined_type = ID ;

other_type = '*array[1..5] of integer*' ;

letter = '*a*' | '*b*' | '*c*' | '*d*' | '*e*' | '*f*' | '*g*' | '*h*' | '*i*' | '*j*' | '*k*' | '*l*' | '*m*' | '*n*' | '*o*' | '*p*' | '*q*' | '*r*' | '*s*' | '*t*' | '*u*' | '*v*' | '*w*' | '*x*' | '*y*' | '*z*' ;

digit = '*0*' | '*1*' | '*2*' | '*3*' | '*4*' | '*5*' | '*6*' | '*7*' | '*8*' | '*9*' ;

instruction_list = instruction , { '*;*' , instruction } ;

instruction = assign_instruction | if_instruction | io_instruction | while_loop ;

assign_instruction = ID , '*:=*' , expression , '*;*' ;

expression = (ID | CONST) , { operator , (ID | CONST) } ;

if_instruction = '*if*' , boolean_expression , '*then begin'* , instruction_list , '*end;*' , ['*else begin'* , instruction_list , '*end;*' ] ;

boolean_expression = ID , bool_operator , expression ;

io_instruction = '*readln(*' , ID , '*);*' | '*writeln(*' , ID , '*);*' ;

while_loop = '*while*' , boolean_expression , ' *do begin* ' , instruction_list , '*end;*' ;

bool_operator = '>' | '<' | '<>'; operator = '+' | '-' | '*' ;

ID = letter , { letter | digit } ;

CONST = digit , ['.'] , { digit } ;

## 2. Textele sursă a 3 mini-programre

1. Perimetrul și aria unui cerc cu rază dată:

```
program cerc;
var a, p, r: real;
begin
	readln(r);
	p := 2*3.14*r;
	a := 3.14 * r *r;
	writeln(p);
	writeln(a);
end
```

2. CMMDC a două numere:

```
program cmmdc;
var a, b: integer;
begin
	readln(a);
	readln(b);
	while a<>b do
	begin
		if a>b then
		begin
			a:=a-b;
		end;
		else begin
			b:=b-a;
		end;
	end;
	writeln(a);
end
```

3. Suma a n numere citite de la tastatură:

```
program suma;
var n, sum, nr: integer;
begin
	readln(n);
	sum := 0;
	while n>0 do
	begin
		readln(nr);
		sum := sum + nr;
		n := n - 1;
	end;
	writeln(sum);
end
```

## 3. Programe ce conțin erori:

1. Două erori care sunt în același timp erori în limbajul original (pentru care MLP definește un subset)

```
program err1;
var a, 2b: integer;
begin
	readln(a)
	writeln(a);
end
```

2. Două erori conform MLP, dar care nu sunt erori în limbajul original.

```
program ex1;
var a, b: word;
begin
	readln(a, b);
	a := a + b;
	writeln(a);
end
```