

## Seminar 1

### 1.1

Avem diagrama de sintaxa si presupunem ca specificarea elementelor lexicale este cunoscuta.

Astfel, consideram ca:

<identificator> este o succesiune de litere si cifre care incepe cu o litera.

<cifra> este una (oricare) dintre: 0,1,2,3,4,5,6,7,8,9

<cifra\_hex> este una (oricare) dintre 0,1,2,3,4,5,6,7,8,9, A, B, C, D, E, F

Exemple pentru prima diagrama:

(abc123)

(a,b,c123)

Exemple pentru a doua diagrama:

+1234

-\$ABCDE

7

### 1.2

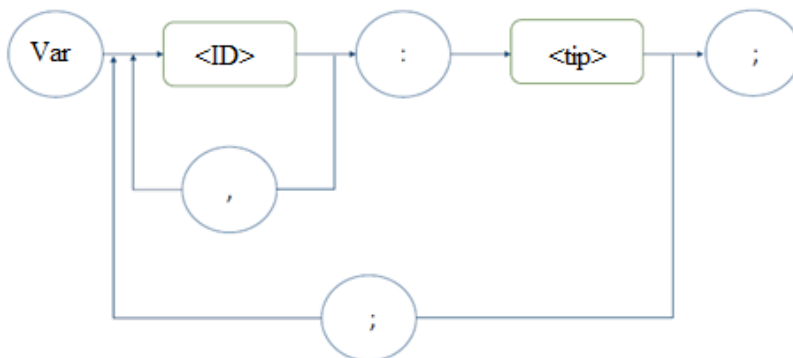
Exemplu:

Var a,b,c: integer;

x: real;

Sectiunea incepe cu cuvantul var , dupa care urmeaza listele de declaratii. O lista de declaratii consta dintr-o lista de variabile separate prin virgula, urmate de ":" apoi de tipul lor si se incheie cu ";"

In locul numelor variabilelor vom folosi <ID>, iar in locul tipului lor vom folosi: <tip>



### 3.1

a)

Terminale:

"begin", "end", ".", "ID", "=", "+", "if", "then", "(", ")", "+"

Neterminale:

<program> , <lista\_instr> , <instr> , <atribuire> , <expr> , <variabila> , <instr\_if>

b)

begin

abc = xy;

if (xy) then abcab=abc

end.

begin

```
aa=ab;
ac=ab+ac
end.
```

### 3.2

Elemente lexicale:

Cuvinte rezervate/cheie: var, integer, begin, end

ID: a1, a2, a3, f

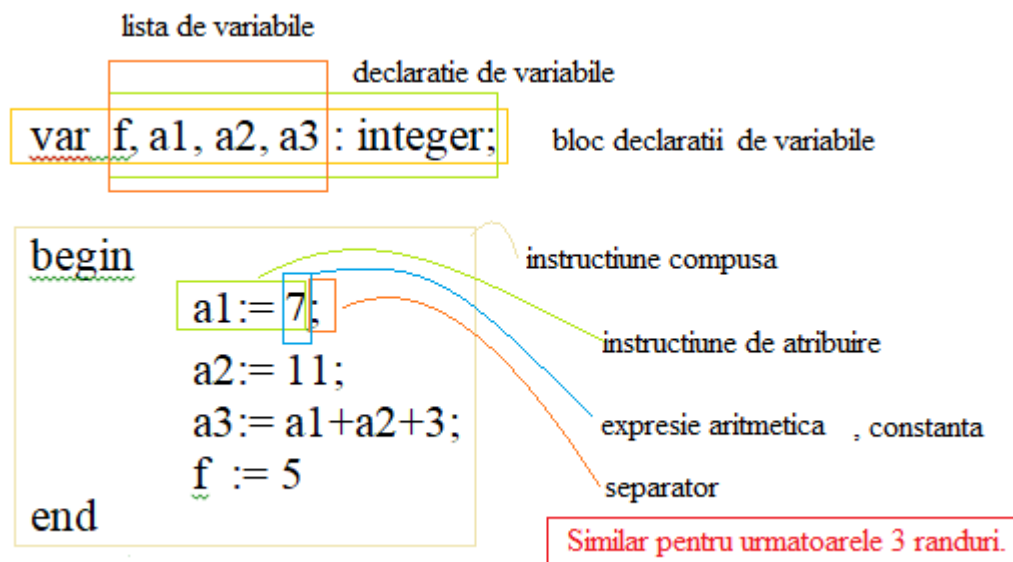
Operatori: ":", "+", ":", "

Delimitatori/separatori: ",", ":", ":", ":", ":", ":", "\n"

CONST: 7, 11, 5, 3

Obs: In pascal, integer nu este cuvint cheie, dar, pentru simplificare, noi il vom considera cuvint cheie. Se poate folosi aceasta simplificare si in cadrul temelor de laborator.

Structuri sintactice:



b) specificare in BNF

```
<program>      ::= <bloc-decl-var> <instr-comp> .
<bloc-decl-var> ::= var <lista-decl-var>
<lista-decl-var> ::= <decl-var> | <lista-decl-var> <decl-var>
<decl-var>      ::= <lista_variabila> : integer;
<lista_variabila> ::= ID , <lista_variabila> | ID
<instr-comp>    ::= begin <lista_instr> end
<lista_instr>    ::= <instr> ; <lista_instr> | <instr>
<instr>         ::= <atribuire>
<atribuire>     ::= ID := <expr>
<expr>          ::= <expr> + <expr> | CONST | ID
```

c)

```
var    a1 : integer;
       a2, a3 : integer;

begin
  a1 := a2 + a3 + 7
```

end.

d) i.

Observatie: ID si CONST sunt atomi lexicali speciali, pentru ei vom avea codurile 0 si 1.

Atom lexical	Cod Atom
ID	0
CONST	1
var	2
,	3
:	4
integer	5
;	6
begin	7
:=	8
+	9
end	10
.	11

Obs.: In FIP am colorat cu galben pozitia corespunzatoare “;” doar cu scopul de a ne urmari mai usor pozitia curenta din program.

FIP	
<u>Cod Atom</u>	<u>COD TS</u>
2	
0	4
3	
0	1
3	
0	2
3	
0	3
4	
5	
6	
7	
0	1
8	
1	3
6	
0	2
8	
1	4
6	
0	3
8	
0	1
9	
0	2
9	
1	1
6	
0	4
8	
1	5
10	
11	

### TS pentru ID

<u>Pozitie in tabel</u>	<u>Simbol (ID)</u>	<u>Alte attribute</u>
1	a1	Se memoreaza si alte attribute (daca e cazul)
2	a2	
3	a3	
4	f	

Pozitia in tabel nu se memoreaza. Poate incepe de la 0 sau de la 1.

### TS pentru CONST

<u>Poz. in tabel</u>	<u>Simbol (CONST)</u>	<u>Alte attribute</u>
1	3	Se memoreaza si alte attribute (daca e cazul)
2	5	
3	7	
4	11	

Observatii:

Tabelele TS pentru identificatori si constante sunt ordonate lexicografic.

Construirea lor in aceasta forma poate ridica unele dificultati. Discutia despre acestea a avut loc la seminar.

O a doua varianta de constructie a acestor tabele este de a folosi campuri de legatura/inlantuire. Sa luam exemplul tabelii identificatorilor. Astfel, la construirea tabelii, noii identificatori vor fi adaugati la sfarsit, dar vom avea o coloana suplimentara in tabel care va indica urmatorul element in ordine lexicografica. In acest caz, tabelul va arata astfel (iar valorile din FIP folosite pentru identificarea atomilor vor fi altele):

### TS pentru ID

<u>Pozitie in tabel</u>	<u>Simbol (ID)</u>	<u>Legatura ordine lexicografica</u>
1	f	-1
2	a1	3
3	a2	4
4	a3	1

Astfel, avem o lista simplu inlantuita ordonata, reprezentata pe tabel, iar pozitia 2 este pozitia capului listei.

In cazul in care optam pentru a reprezenta inlantuit pe tabel arborele binar de cautare, tabelul va arata astfel:

### TS pentru ID

<u>Pozitie in tabel</u>	<u>Simbol (ID)</u>	<u>Legatura stanga</u>	<u>Legatura dreapta</u>
1	f	2	-1
2	a1	-1	3
3	a2	-1	4
4	a3	-1	-1

iii) Pentru reprezentare folosind tabela de dispersie, trebuie sa alegem o functie de dispersie si o strategie de rezolvare a coliziunilor.

Fie:

$m$ =dimensiunea tabelii de dispersie

functia de dispersie = (codul ascii al primului caracter) mod  $m$

cu: open addressing & linear probing

Daca  $m=11$ , tabela de simboluri pentru identificatori va fi:

<u>Pozitie in tabel</u>	<u>Simbol (ID)</u>
0	a3
1	
2	

3	f
4	
5	
6	
7	
8	
9	a1
10	a2