

STRUCTURI

8.1. ASPECTE TEORETICE

Utilizatorul recurge la folosirea structurilor atunci când dorește să grupeze date de tipuri diferite și să le prelucereze atât individual cât și împreună.

Observație. *Structura* este compusă dintr-un număr de componente de *anumite tipuri*. Componentele structurii se numesc *câmpuri* sau *membri*.

Observație. Fiecare câmp trebuie să aparțină unui tip *deja definit* sau unui tip *standard*.

Observație. Sintaxa declarării unei structuri este:

```
struct [nume_structura] {
    tip_1 câmp_1;
    tip_2 câmp_2;
    .....
    tip_n câmp_n;
}[lista_variabale_structura];
```

♦ **Exemplul 1.** Se definește structura *elev* care are trei câmpuri.

```
struct elev{
    char nume[30];
    char prenume[30];
    float media;
};
```

♦ **Exemplul 2.** Se definește structura *masina* care are trei câmpuri.

```
struct masina{
    char nume[30];
    char culoare[30];
    float marca;
};
```

Observație. Numele structurii și lista variabilelor, întâlnite în sintaxa prezentată mai sus, pot lipsi însă **nu ambele simultan**. În cazul când numele structurii lipsește spunem că variabilele au un **tip anonim**

◆ **Exemple de declarații corecte:**

```
struct {
    char nume[30 ];
    char prenume[30 ];
    float media;
} v1, v1,...,vn;
```

sau:

```
struct elev{
    char nume[30 ];
    char prenume[30 ];
    float media;
};
```

elev v1, v2, ..., vn;

sau:

```
struct elev{
    char nume[30 ];
    char prenume[30 ];
    float media;
};
```

struct elev v1, v2, ..., vn;

- Observații.**
1. După acolada '}' se pune obligatoriu ';' 'dacă nu se declară nici o variabilă.
 2. După fiecare linie a structurii se pune obligatoriu ';' '.
 3. Pot exista mai multe câmpuri de același tip.

◆ **Exemple de declarații corecte:**

1.

```
struct elev{
    char nume[30 ];
    char prenume[30 ];
    float media;
};
```

2.

```
struct elev{
    char nume[30 ], prenume[30 ];
    float media, greutatea;
};
```

♦ Exemple de declarații incorecte:

1.

```
struct elev{
    char nume[30];
    char prenume[30];
    float media;
}
```

Nu s-a pus ';' după '}'.

2.

```
struct elev{
    char nume[30];
    char prenume[30];
    float media;
};
```

S-au pus ';' după *un câmp*.

Observație. Sintaxa de definire a unui tip structură:

```
typedef struct{
    tip_1 câmp_1;
    tip_2 câmp_2;
    .....
    tip_n câmp_n;
}nume_tip;
```

♦ Exemplu de definiție și declarație corectă:

```
typedef struct {
    char nume[30];
    char prenume[30];
    float media;
} elev;
```

elev v1, v2, ..., vn;

Accesarea unui câmp al unei variabile *a* de tip structură, se face astfel:

```
a.câmp_1
a.câmp_2
.....
```

(se pune '.' (punct) între numele variabilei și numele câmpului accesat).

♦ Exemplu.

Fie:

```
typedef struct {
    char nume[30];
    char prenume[30];
    float media;
} elev;
```

elev S;

- câmpul *nume* al variabilei *S* se accesează : *S.nume;*
- câmpul *prenume* al variabilei *S* se accesează : *S.prenume;*
- câmpul *media* al variabilei *S* se accesează : *S.media.*

Secvențele de citire a unei variabile de tipul *structură*:

```
Struct nume_structura {
    tip_1 câmp_1;
    tip_2 câmp_2;
    .....
    tip_n câmp_n;
}a;
```

Metoda 1:

```
cin>>a.câmp_1;
cin>>a.câmp_2;
.....
cin>>a.câmp_n;
```

Metoda2:

```
cin>>a.câmp_1>>...>>a.câmp_n;
```

Se citește fiecare câmp în parte (mai sunt și alte metode decât cele prezentate).

◆ Exemplu.

Fie:

```
typedef struct {
```

```
    char nume[30];
    char prenume[30];
    float media;
} elev;
```

```
elev S;
```

◆ **Metoda 1:**

```
cout<<" Numele      : ";  cin>>S.nume;
cout<<" Prenumele   : ";  cin>>S.prenume;
cout<<" Media       : ";  cin>>S.media;
```

◆ **Metoda 2:**

```
cin>>S.nume>>S.prenume>>S.media;
```

Secvențele de afișare a unei variabile de tipul *structură*:

Struct *nume_structura* {

tip_1 câmp_1;

tip_2 câmp_2;

.....
tip_n câmp_n;

} a;

Metoda 1:

cout<<a.câmp_1<<endl;

cout<<a.câmp_2<<endl;

.....
cout<<a.câmp_n<<endl;

Metoda2:

cout<<a.câmp1<<" "<<...<<a.câmpn;

Se afișează fiecare câmp în parte (mai sunt și alte metode decât cele prezentate).

◆ Exemplu.

typedef struct {

char nume[30];

char prenume[30];

float media;

} elev;

elev S;

◆ Metoda 1:

cout<<" Numele : "; cout<<S.nume<<endl;

cout<<" Prenumele : "; cout<<S.prenume<<endl;

cout<<" Media : "; cout<<S.media<<endl;

◆ Metoda 2:

cout<<S.nume<<" "<<S.prenume<<" "<<S.media;

Observație. Este posibil să se declare un tablou cu elemente de tipul structură.

◆ Exemplu.

typedef struct {

char nume[30];

char prenume[30];

float media;

} elev;

elev S[20];

◆ Citirea a n componente ale vectorului S se poate face:

◆ Metoda 1:

```
cout<<"n="; cin>>n;
for (i=1;i<=n;i++)
{
    cout<<"Numele      : "; cin>>S[i].nume;
    cout<<"Prenumele   : "; cin>>S[i].prenume;
    cout<<"Media       : "; cin>>S[i].media;
}
```

◆ Metoda 2:

```
cout<<"n="; cin>>n;
for (i=1;i<=n;i++)
    cin>>S[i].nume>>S[i].prenume>>S[i].media;
```

◆ Afișarea a n componente ale vectorului S se poate face:

◆ Metoda 1:

```
for (i=1;i<=n;i++)
{
    cout<<"Numele      : "; cout<<S[i].nume<<endl;
    cout<<"Prenumele   : "; cout<<S[i].prenume<<endl;
    cout<<"Media       : "; cout<<S[i].media<<endl;
    cout<<endl;
}
```

◆ Metoda 2:

```
for (i=1;i<=n;i++)
    cout<<S[i].nume<<" " <<S[i].prenume<<" " <<S[i].media<<endl;
```

Observație. Este posibilă definirea unei structuri ale cărei componente sunt de tipul structură, definită anterior.

◆ Exemplu.

```
struct adresa{
    char strada[30 ],
    char cartier[30 ];
};

typedef struct {
    char nume[30 ];
    float media;
```

adresa detalii;
} elev;

elev S;

♦ Accesarea componentelor se face:

S.num

S.detalii.strada

S.detalii.cartier

Inițializarea structurilor

Observație. Componentele unei structuri se pot inițializa. Pentru aceasta la sfârșitul declarației sau definiției structurii se pune caracterul '=' și se scriu între acolade, în ordine valorile componentelor separate prin virgule.

♦ Exemplu.

```
struct adresa{
```

```
    char strada[30];
```

```
    char cartier[30];
```

```
};
```

```
typedef struct {
```

```
    char nume[30];
```

```
    float media;
```

```
    adresa detalii;
```

```
} elev;
```

```
elev S={"Popescu",7.61,{"Pionierului","Rovine"}};
```

Observație. Ultimele valori ale componentelor pot lipsi. Câmpurile cărora nu le corespund valori se inițializează automat astfel:
câmpurile numerice cu 0;
câmpurile șiruri de caractere cu șirul vid.

```
typedef struct {
```

```
    char nume[30];
```

```
    char prenume[30];
```

```
    float media;
```

```
} elev;
```

```
elev S={"Popescu"};
```

După această inițializare:

S.num="Popescu"

S.prenume="\0";

S.media=0.