

Tipul struct

Tipuri structurate de date. Tipul struct

- ▶ Aspecte teoretice
- ▶ Declararea structurilor
- ▶ Declararea variabilelor de tip struct
- ▶ Accesarea câmpurilor structurii
- ▶ Operații elementare
- ▶ Structuri imbricate
- ▶ Utilitatea structurilor
- ▶ Implementări sugerate
- ▶ Probleme propuse
- ▶ Bibliografie

▶ Aspecte teoretice

- tipurile de date se clasifică în două categorii:
 - o tipuri simple (standard/elementare, exp. *int*, *float*, *char*, etc.)
 - o tipuri compuse (structurate, exp. tablou, înregistrare, etc.)
- tipul struct (tipul înregistrare/articol)
- o structură (înregistrare) este o colecție de valori eterogene (neomogene) ca tip, stocate într-o zonă compactă de memorie
- o structură este un nou tip de date definit de programator care permite gruparea unor date (de regulă având tipuri diferite) sub un singur nume
- o structură este formată dintr-un număr fix sau variabil de componente numite *câmpuri* care sunt identificate prin nume simbolice, denumite *selectori*

Exemplu:

Pentru memorarea informațiilor despre elevii unei clase (numărul matricol, numele, prenumele, media) se poate utiliza o structură de date de tip înregistrare. Fiecare atribut care descrie elevul va reprezenta un câmp al înregistrării.

Structura elev

Nume câmp	Tip câmp	Valoare câmp
nr_matricol	întreg	125
nume	șir de caractere	Pop
prenume	șir de caractere	Ana
media	real	9,5

▶ Declararea structurilor

Sintaxa:

```
struct [<nume_structură>]
{
    <tip_dată1> <nume11>, <nume12>, ..., <nume1n>;
    <tip_dată2> <nume21>, <nume22>, ..., <nume2n>;
    . . . . .
    <tip_datăm> <numem1>, <numem2>, ..., <numemn>;
} [<lista_variabile>];
```

Tipul struct

unde:

- <nume_structură> este identificatorul structurii
- <tip_dată_i> reprezintă tipurile de date pentru câmpurile structurii
- <nume_{ij}> reprezintă identificatori de câmpuri
- <lista_variabile> reprezintă variabile de tip struct

Observații:

- <nume_structură> sau <lista_variabile> pot lipsi dar nu simultan
- definirea structurii se poate face în funcția *main()*, dar este recomandabil de făcut înaintea funcției *main()*

Exemplu:

```
struct elev
{
    int nr_matricol;
    char nume[21], prenume[21];
    float media;
};
```

► Declararea variabilelor de tip struct

Sintaxa:

```
[struct] <nume_structură> <lista_variabile>;
```

Observații:

- o variabilă de tip struct se poate declara după declararea tipului struct sau se poate declara simultan cu tipul struct
- o variabilă de tip struct poate fi inițializată la declarare
- cuvântul cheie *struct* este opțional

Exemple:

1.

```
struct elev e1, e2, e[30];
```

2.

```
elev e1, e2, e[30];
```

3.

```
struct //tip structură anonim
{
    int nr_matricol;
    char nume[21], prenume[21];
    float media;
}e1,e2;
```

4.

```
elev e1={125,"Pop","Ana",9.5};
```

► Accesarea câmpurilor structurii

Sintaxa:

```
<nume_variabilă_struct>.<nume_câmp>
```

Tipul struct

Observații:

- un câmp al structurii se poate folosi doar dacă numele câmpului este precedat de numele variabilei structură din care face parte

Exemple:

```
e1.nr_matricol, e1.num, e1.prenume, e1.media  
e2.nr_matricol, e2.num, e2.prenume, e2.media
```

▶ Operații elementare

a. atribuire

- unei variabile de tip struct i se poate atribui o altă variabilă de tip struct
- atribuirea între două variabile de tip struct se numește *copiere bit cu bit*

Exemplu:

```
e1=e2;
```

b. citirea unei variabile de tip struct

Exemplu:

```
cin>>e1.nr_matricol;  
cin>>e1.num;  
cin>>e1.prenume;  
cin>>e1.media;
```

sau

```
cin>>e1.nr_matricol>>e1.num>>e1.prenume>>e1.media
```

c. afișarea unei variabile de tip struct

Exemplu:

```
cout<<e1.nr_matricol<<' '  
cout<<e1.num<<' '  
cout<<e1.prenume<<' '  
cout<<e1.media;
```

sau

```
cout<<e1.nr_matricol<<' '<<e1.num<<' '<<e1.prenume<<' '<<e1.media;
```

▶ Structuri imbricate

- o structură poate conține la rândul ei componente (câmpuri) care sunt de tip structură
- câmpurile de tip structură se numesc *structuri imbricate* (incluse)

Exemplu:

Pentru a memora pentru elev data nașterii (ziua, luna și anul) se poate utiliza o structură de date de tip înregistrare inclusă în structura elev.

Structura data_n

Nume câmp	Tip câmp	Valoare câmp
zi	întreg	12
luna	șir de caractere	mai
an	întreg	2005

Tipul struct

Structura elev

Nume câmp	Tip câmp	Valoare câmp	
nr_matricol	întreg	125	
nume	șir de caractere	Pop	
prenume	șir de caractere	Ana	
media	real	9,5	
data_n	data_nașterii	întreg	12
		șir de caractere	mai
		întreg	2005

Exemplu:

```
struct data_nasterii
{
    int zi;
    char luna[15];
    int an;
};
struct elev
{
    int nr_matricol;
    char nume[21], prenume[21];
    float media;
    data_nasterii data_n;
}e;
```

sau

```
struct elev
{
    int nr_matricol;
    char nume[21], prenume[21];
    float media;
    struct
    {
        int zi;
        char luna[15];
        int an;
    }data_n;
}e;
```

Accesarea câmpurilor:

```
e.data_n.zi, e.data_n.luna, e.data_n.an
```

► Utilitatea structurilor

- programele devin mai explicite
- pot fi definite tipuri de date specifice fiecărei aplicații
- poate fi redus numărul de parametri al unor funcții

► Implementări sugerate

1. Se citește de pe prima linie a fișierul text `elevi.in` un număr natural n ($1 \leq n \leq 30$), iar apoi se citesc de pe fiecare din următoarele n linii datele câte unui elev: numărul matricol - număr natural, numele - cel mult 20 de caractere, prenumele - cel mult 20 de caractere, media - număr real și data nașterii sub forma zi - număr natural, lună - cel mult 14 caractere, an - număr natural. Să se afișeze pe câte o linie a fișierului text `elevi.out` datele citite pentru fiecare elev, separate prin câte un spațiu.

Tipul struct

Exemplu:

elevi.in	elevi.out
3	101 Pop Ana 9.5 12 ianuarie 2010
101 Pop Ana 9.5 12 ianuarie 2010	102 Adam Ionel 10 21 mai 2009
102 Adam Ionel 10 21 mai 2009	103 David Luca 8.5 10 august 2011
103 David Luca 8.5 10 august 2011	

```
#include <fstream>

using namespace std;

ifstream fin("elevi.in");
ofstream fout("elevi.out");

struct elev
{
    int nr_matricol;
    char nume[21], prenume[21];
    float media;
    struct
    {
        int zi;
        char luna[15];
        int an;
    }data_n;
};

int main()
{
    int n,i;
    elev e[31];
    fin>>n;
    for(i=1;i<=n;i++)
    {
        fin>>e[i].nr_matricol;
        fin>>e[i].nume;
        fin>>e[i].prenume;
        fin>>e[i].media;
        fin>>e[i].data_n.zi;
        fin>>e[i].data_n.luna;
        fin>>e[i].data_n.an;
    }
    for(i=1;i<=n;i++)
    {
        fout<<e[i].nr_matricol<<' ';
        fout<<e[i].nume<<' ';
        fout<<e[i].prenume<<' ';
        fout<<e[i].media<<' ';
        fout<<e[i].data_n.zi<<' ';
        fout<<e[i].data_n.luna<<' ';
        fout<<e[i].data_n.an<<'\n';
    }
    fin.close();
    fout.close();
    return 0;
}
```

2. Se citește din fișierul text `date.in` un număr natural n ($1 < n < 10^5$). Să se descompună numărul n în factori primi, memorând rezultatul sub forma unui vector de structuri cu doua câmpuri: unul care conține divizorii primi, iar celălalt care conține puterile la care apar divizorii

Tipul struct

primi în descompunere. Să se afișeze pe câte o linie a fișierului `date.out` divizorul prim și puterea acestuia despărțite prin câte un spațiu.

Exemplu:

<code>date.in</code>	<code>date.out</code>
8500	2 2 5 3 17 1

```
#include <fstream>

using namespace std;

ifstream fin("date.in");
ofstream fout("date.out");

struct factori
{
    int divizor,exponent;
};

int main()
{
    int n,nr,d,p,i;
    factori f[1001]={0};
    fin>>n;
    p=0;
    nr=0;
    while (n%2==0)
    {
        p++;
        n=n/2;
    }
    if (p)
    {
        nr++;
        f[nr].divizor=2;
        f[nr].exponent=p;
    }
    d=3;
    while (n!=1)
    {
        if (n%d==0)
        {
            p=0;
            while (n%d==0)
            {
                p++;
                n=n/d;
            }
            nr++;
            f[nr].divizor=d;
            f[nr].exponent=p;
        }
        d=d+2;
        if (d*d>n)
            d=n;
    }
    for (i=1;i<=nr;i++)
    {
```

Tipul struct

```
fout<<f[i].divizor<<' '<<f[i].exponent<<'\n';  
}  
fin.close();  
fout.close();  
return 0;  
}
```

► Probleme propuse

Pbinfo:

- ✘ FrațiiMax - 923
- ✘ Aniversări - 1013
- ✘ Serbare - 1460

► Bibliografie

1. Miloșescu M., *Informatică. Manual pentru clasa a X-a*, Editura Didactică și Pedagogică, București, 2005
2. Cerchez E., Șerban M., *Programarea în limbajul C/C++ pentru liceu*, Editura Polirom, Iași, 2005
3. Popescu C., *Culegere de probleme de informatică*, Editura Donaris-Info, Sibiu, 2002
4. Ministerul Educației, Cercetării și Tineretului, Centrul Național pentru Curriculum și Evaluare în Învățământul Preuniversitar, *Proba scrisă la informatică. Examenul de bacalaureat – Variante (1-100)*, București 2008
5. [http://en.wikipedia.org/wiki/Struct_\(C_programming_language\)](http://en.wikipedia.org/wiki/Struct_(C_programming_language))
6. <http://www.cplusplus.com/doc/tutorial/structures/>
7. www.ududec.com