

# Introducere în programare 2016 - 2017

5

Bogdan Pătruț

[bogdan@info.uaic.ro](mailto:bogdan@info.uaic.ro)

# Curs 5: Structuri dinamice de date

- Liste simplu înlănțuite
- Liste dublu înlănțuite
- Stive
- Cozi
- Arbori binari

# Liste simplu înlănțuite

(LIFO, declarație simplă)

```
#define tipDate int
```

```
struct nod
```

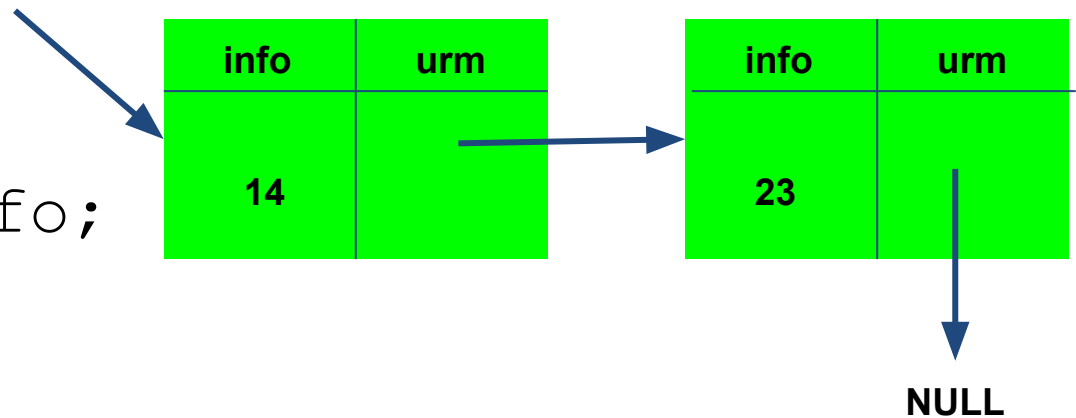
```
{
```

```
    tipDate info;
```

```
    nod* urm;
```

```
};
```

```
typedef nod* listaSimpla;
```



# Citirea unei liste simplu înlănțuită

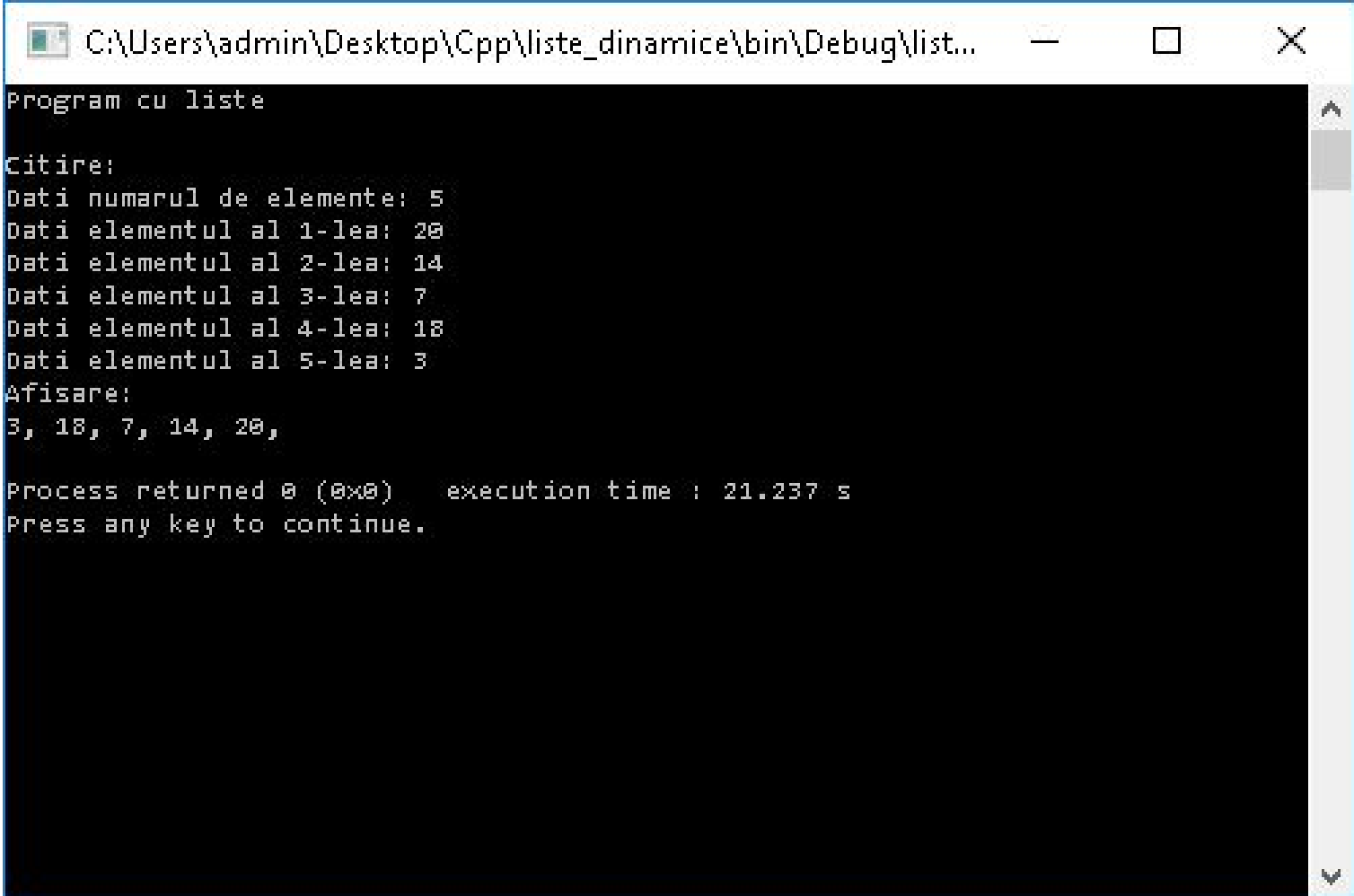
```
void citesteListaSimpla(listaSimpla& L) {  
    unsigned int nrElemente;  
    cout<<"Dati numarul de elemente: ";  
    cin>>nrElemente;  
    tipDate element;  
    L=NULL; listaSimpla p;  
    for (unsigned int i=1; i<=nrElemente; i++)  
    {  
        cout<<"Dati elementul al "<<i<<"-lea: ";  
        cin>>element; p=new nod; p->info=element;  
        p->urm=L;L=p;  
    }  
}
```

# Afişarea unei liste simplu înlănţuite

```
void afiseazaListaSimpla(listaSimpla L) {
    listaSimpla p; p=L;
    while (p!=NULL) {
        cout<<p->info<<" ";
        p=p->urm;
    }
    cout<<"\n";
}

int main() {
    cout<<"Program cu liste\n\n"; listaSimpla L;
    cout<<"Citire:\n"; citesteListaSimpla(L);
    cout<<"Afisare:\n"; afiseazaListaSimpla(L);
    return 0; }
```

# Execuția programului



```
C:\Users\admin\Desktop\Cpp\liste_dinamice\bin\Debug\list...
Program cu liste
Citire:
Dati numarul de elemente: 5
Dati elementul al 1-lea: 20
Dati elementul al 2-lea: 14
Dati elementul al 3-lea: 7
Dati elementul al 4-lea: 18
Dati elementul al 5-lea: 3
Afisare:
3, 18, 7, 14, 20,

Process returned 0 (0x0)   execution time : 21.237 s
Press any key to continue.
```

# Listă dublu înlănțuită (structură, FIFO)

```
#define tipDate int
struct nod {
    tipDate info;
    nod* urm;
    nod* prec;
};
struct listaDubla {
    nod* prim; nod* ultim;
    unsigned int lungime;
};
```

# Inițializarea listei

```
void initializeazaListaDubla (listaDubla&
    L)
{
    L.lungime=0;
    L.prim=NULL; L.ultim=NULL;
}
```



# Adăugarea unui element la listă

```
void adaugaLaListaDublaElement(listaDubla& L,  
    tipDate element) {  
    if (L.lungime==0) {  
        L.prim=new nod; L.prim->info=element;  
        L.prim->urm=NULL; L.prim->prec=NULL;  
        L.ultim=L.prim; L.lungime++;  
    }  
    else {  
        nod* p; p=new nod;  
        p->info=element; p->urm=NULL;  
        p->prec=L.ultim; L.lungime++;  
        L.ultim->urm=p; L.ultim=p;  
    }  
}
```

# Citirea listei dublu înlănțuite

```
void citesteListaDubla(listaDubla& L)
{
    unsigned int nrElemente;
    cout<<"Dati numarul de elemente: ";
    cin>>nrElemente;
    tipDate element;
    for (unsigned int i=1; i<=nrElemente; i++)
    {
        cout<<"Dati elementul al "<<i<<"-lea:
";
        cin>>element;
        adaugaLaListaDublaElement(L, element);
    }
}
```

# Afișarea listei dublu înlănțuite

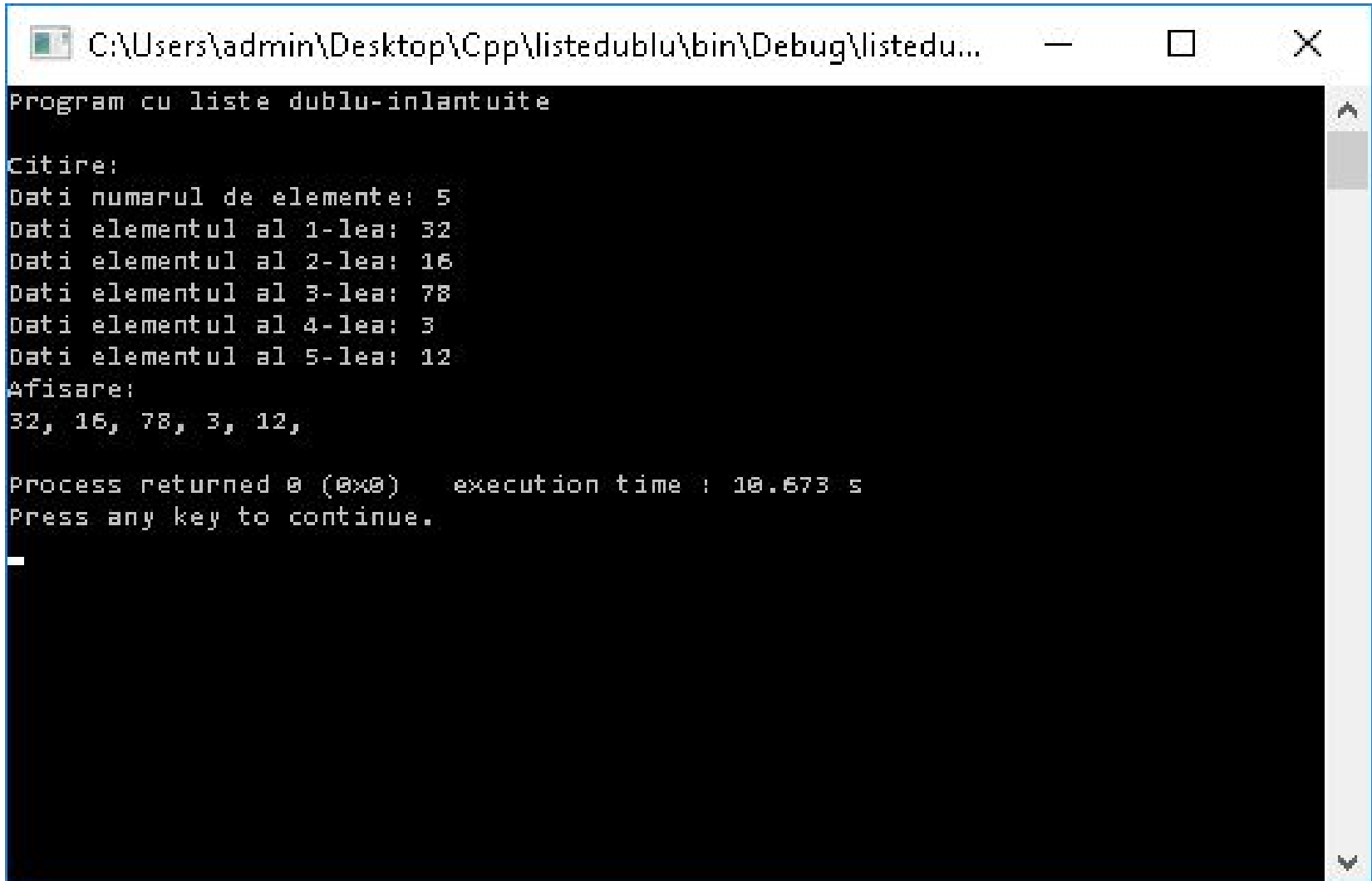
```
void afiseazaListaDubla (listaDubla L)
{
    nod* p=L.prim;
    while (p!=NULL)
    {
        cout<<p->info<<" , ";
        p=p->urm;
    }
    cout<<"\n";
}
```

# Funcția main()

```
#include <iostream>
using namespace std;
#define tipDate int

int main()
{
    cout<<"Program cu liste dublu-inlantuite\n\n";
    listaDubla L;
    initializeazaListaDubla(L);
    cout<<"Citire:\n";
    citesteListaDubla(L);
    cout<<"Afisare:\n";
    afiseazaListaDubla(L);
    return 0;
}
```

# Execuția programului



```
C:\Users\admin\Desktop\Cpp\listedublu\bin\Debug\listedu...
Program cu liste dublu-inlantuite

Citire:
Dati numarul de elemente: 5
Dati elementul al 1-lea: 32
Dati elementul al 2-lea: 16
Dati elementul al 3-lea: 78
Dati elementul al 4-lea: 3
Dati elementul al 5-lea: 12
Afisare:
32, 16, 78, 3, 12,

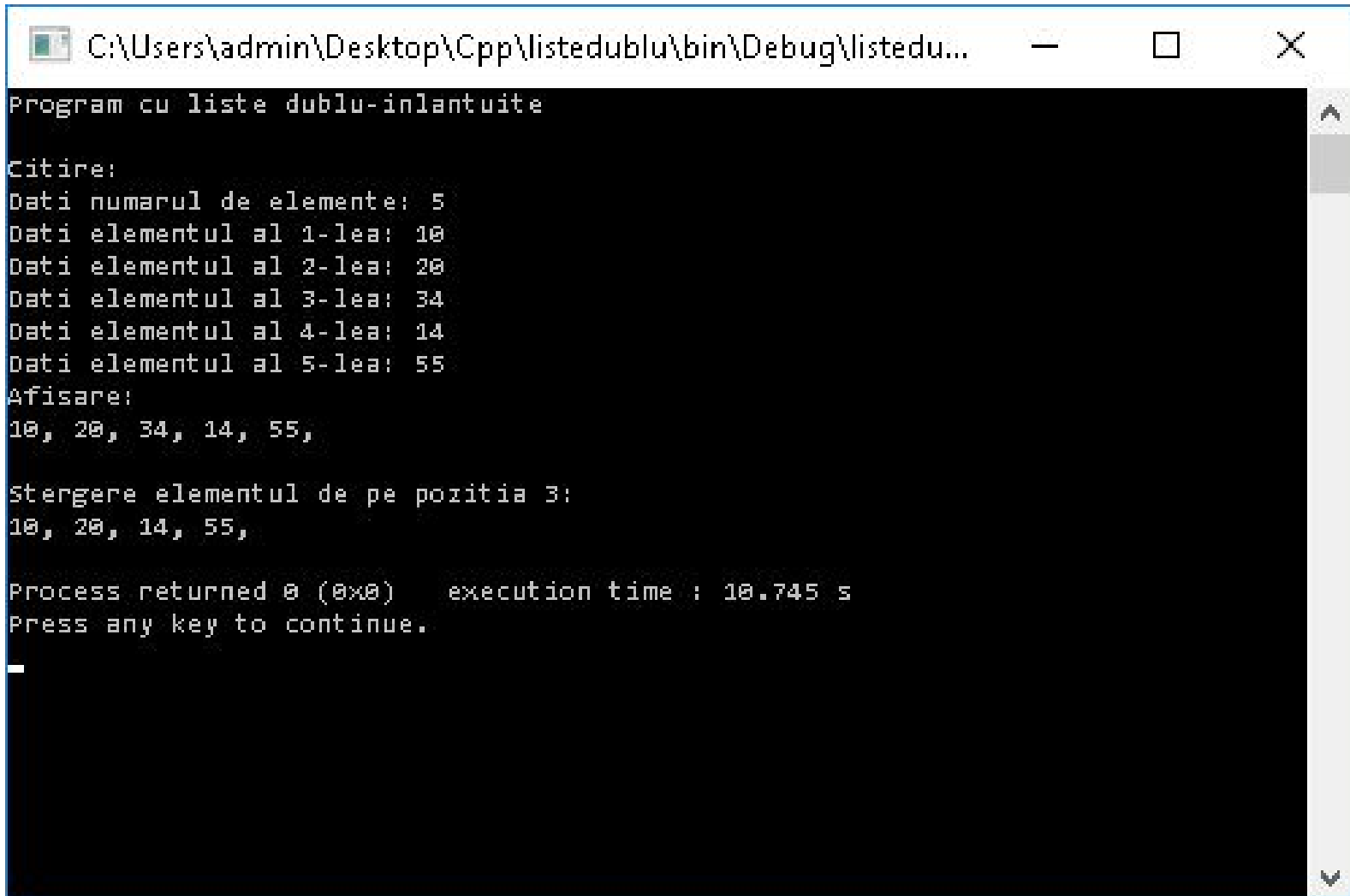
Process returned 0 (0x0)   execution time : 10.673 s
Press any key to continue.
_
```

# Ștergerea unui element

```
void stergeElement(listaDubla L, unsigned int
    pozitieElement) {

    // sterge un element de pe o pozitie din interior
    nod* p=L.prim;
    nod* q; nod* r;
    for (unsigned int i=1; i<=pozitieElement-1;
        i++)
        p=p->urm;
    q=p->prec; r=p->urm;
    q->urm=r; r->prec=q;
    delete p;
    L.lungime--;
}
```

# Execuția programului



```
C:\Users\admin\Desktop\Cpp\listedublu\bin\Debug\listedu...
Program cu liste dublu-inlantuite

Citire:
Dati numarul de elemente: 5
Dati elementul al 1-lea: 10
Dati elementul al 2-lea: 20
Dati elementul al 3-lea: 34
Dati elementul al 4-lea: 14
Dati elementul al 5-lea: 55
Afisare:
10, 20, 34, 14, 55,

Stergere elementul de pe pozitia 3:
10, 20, 14, 55,

Process returned 0 (0x0)   execution time : 10.745 s
Press any key to continue.
_
```

# Arbori binari de căutare

- Să se scrie o structură de date eficientă pentru lucrul cu arborii binari (conținând în noduri numere întregi).
- Să se proiecteze o aplicație simplă care să prelucreze arborii binari



# Proprietate

Fiecare nod conține un element care este mai mare decât elementul din oricare nod al subarborelui stâng (daca există) și mai mic sau egal cu orice element din subarborele drept (dacă există).

# Adăugarea recursivă a unui nou element într-un arbore binar de căutare

- dacă arborele este NULL, atunci se creează un nod în care se pune acest element;
- dacă arborele nu este NULL, atunci:
  - dacă elementul din rădăcină este  $>$  elementul nou sosit, acesta se adaugă în subarborele din stângă;
  - dacă nu, el se adaugă în subarborele din dreapta.

# Declarare arbori binari (de căutare)

```
typedef int tipDate;
```

```
struct nod {  
    tipDate info;  
    struct nod *st; struct nod *dr;  
};
```

```
typedef struct nod * arbore;
```

# Arbori binari de căutare

```
int esteArboreNul(arbore a)
{
    return (a==NULL);
}
```

```
void initArbore(arbore& a)
{
    if (!esteArboreNul(a)) a=NULL;
}
```

# Arbori binari de căutare- adăugarea unui element

```
bool adaugaLaArboreElement(arbore& a, tipDate el) {  
    if (esteArboreNul(a)) {  
        a=new nod;  
        if (!a) return false;  
        a->info = el; a->st = NULL; a->dr = NULL;  
        return true;  
    }  
    else if (el < a->info)  
        return adaugaLaArboreElement(a->st, el);  
    else  
        return adaugaLaArboreElement(a->dr, el);  
}
```

# Căutarea unui element în acest arbore

```
bool existaElementInArbore(arbore a, tipDate el)
{
    if (!esteArboreNul(a)) // sau if (a)
    {
        if (a->info==el) return true;
        else
            if (el<a->info)
                return existaElementInArbore(a->st, el);
            else
                return existaElementInArbore(a->dr, el);
    }
    else
        return false;
}
```

# Preordine: rădăcina, subarborele stând, subarborele drept

```
void parcurgereInPreordine (arbore a)
{
    if (!esteArboreNul (a) )
    {
        cout<<a->info<<" , ";
        parcurgereInPreordine (a->st) ;
        parcurgereInPreordine (a->dr) ;
    }
}
```

# Postordine: subarborele stând, subarborele drept, rădăcina

```
void parcurgereInPostordine (arbore a)
{
    if (!esteArboreNul (a) )
    {
        parcurgereInPostordine (a->st) ;
        parcurgereInPostordine (a->dr) ;
        cout<<a->info<<" , ";
    }
}
```



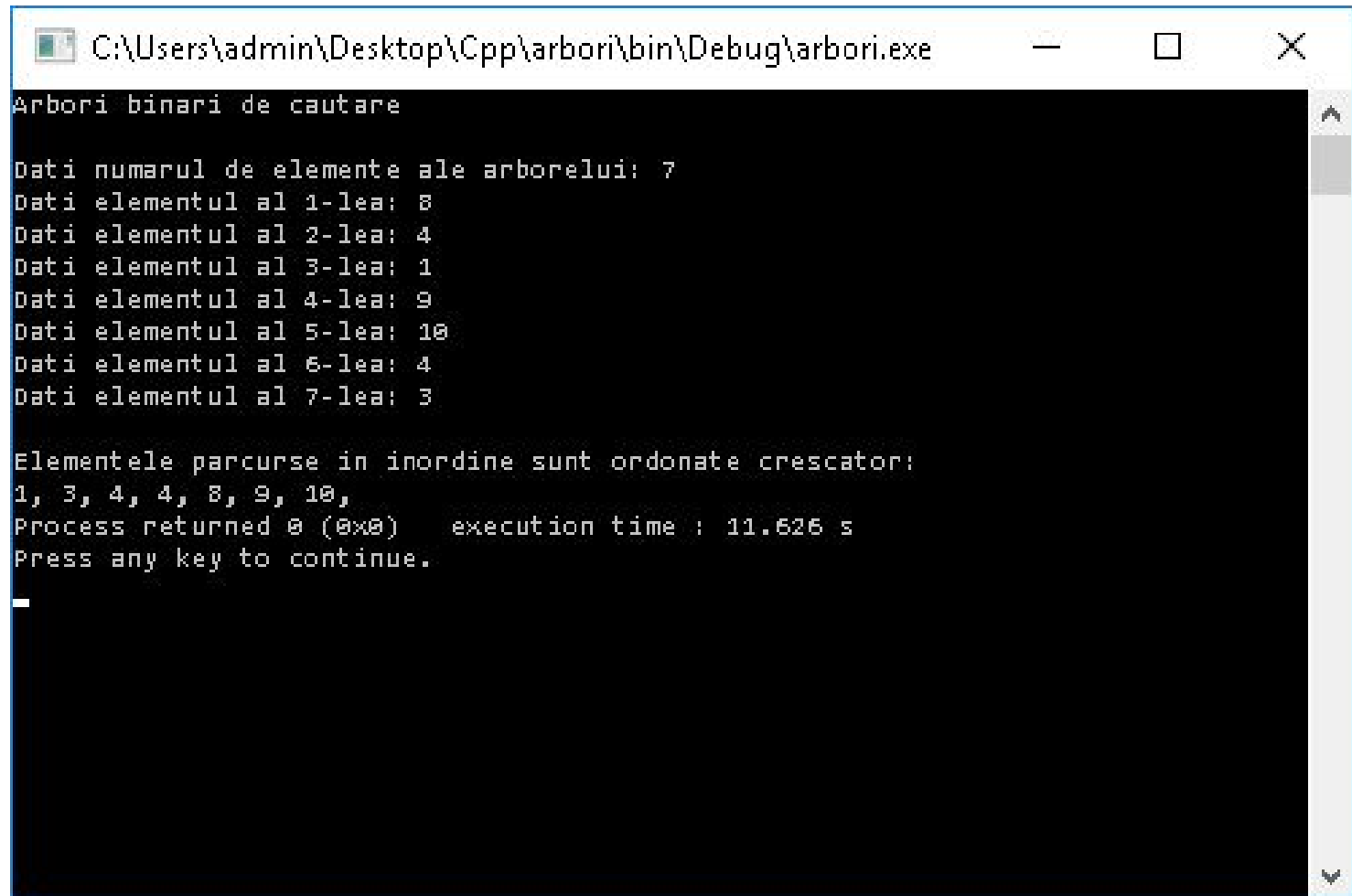
# Inordine: subarborele stâng, rădăcina, subarborele drept

```
void parcurgereInInordine (arbore a)
{
    if (!esteArboreNul (a) )
    {
        parcurgereInInordine (a->st) ;
        cout<<a->info<<" , ";
        parcurgereInInordine (a->dr) ;
    }
}
```

# Un program demonstrativ

```
int main() {
    cout << "Arbori binari de cautare\n\n";
    arbore a; initArbore(a);
    unsigned int i; unsigned int n;
    tipDate x;
    cout<<"Dati numarul de elemente ale arborelui: ";
    cin>>n;
    for (i=1; i<=n; i++) {
        cout<<"Dati elementul al "<<i<<"-lea: ";
        cin>>x; adaugaLaArboreElement(a,x); }
    cout<<"\nElementele parcurse in inordine sunt
ordonate crescator:\n";
    parcurgereInInordine(a);
    return 0; }
```

# Execuția programului



A screenshot of a Windows command prompt window. The title bar at the top reads "C:\Users\admin\Desktop\Cpp\arbori\bin\Debug\arbori.exe" and includes standard minimize, maximize, and close buttons. The command prompt has a black background with white text. The text displayed is as follows:

```
Arbori binari de cautare  
  
Dati numarul de elemente ale arborelui: 7  
Dati elementul al 1-lea: 8  
Dati elementul al 2-lea: 4  
Dati elementul al 3-lea: 1  
Dati elementul al 4-lea: 9  
Dati elementul al 5-lea: 10  
Dati elementul al 6-lea: 4  
Dati elementul al 7-lea: 3  
  
Elementele parcurse in inordine sunt ordonate crescator:  
1, 3, 4, 4, 8, 9, 10,  
Process returned 0 (0x0)   execution time : 11.626 s  
Press any key to continue.  
_
```