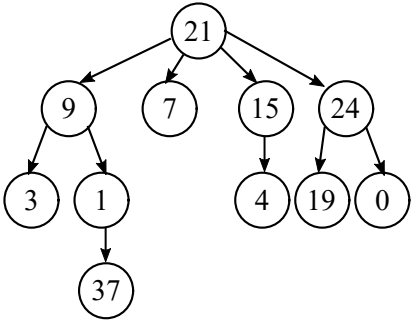
Arbori oarecare. Arbori binari

Universitatea "Transilvania" din Braşov

14 martie 2022

Arbore rădăcină

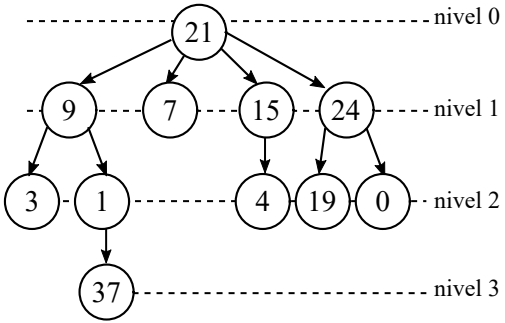


Arbore: În informatică un arbore este o structură ierarhică de noduri, conectate între ele.

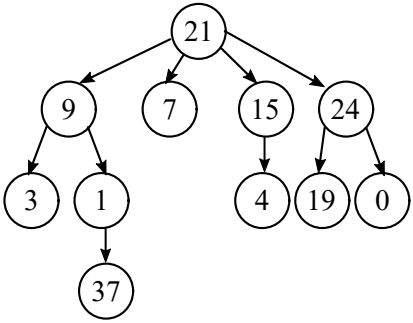
- unul dintre noduri - rădăcină.
- nodurile sunt conectate între ele într-o relație ierarhică părinte - fiu.
- orice nod are un părinte cu excepția rădăcinii.
- orice nod poate avea 0 sau mai mulți fii.
- niciun nod nu împarte vre-un fiu cu alt nod

Din punct de verere al grafurilor, un arbore este un graf conex, fără cicluri.

Arbore rădăcină



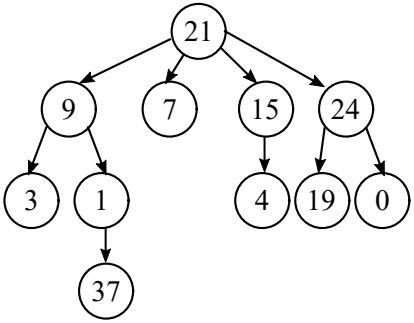
Arbore rădăcină



Definiții: Fie un arbore de rădăcină r și x, y noduri .

- **părinte** al lui x = nodul y pentru care (y, x) muchie.
- **fiu** al lui x = nodul y pentru care (x, y) muchie.
- **strămoș / ascendent** al lui x = un nod y la care pot ajunge de la x urcând de la părinte la părinte
- **urmaș / desscendent** al lui x = un nod y la care pot ajunge printr-o succesiune de fii
- **frați** = doi fii ai aceluiaș nod.
- **frunză**= un nod fără fii
- **nod intern** = un nod care nu este frunză.

Arbore rădăcină

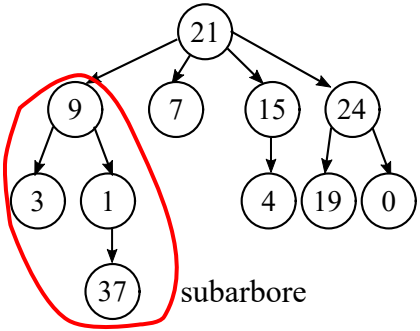


Definiție - adâncimea / înălțimea unui arbore: Lungimea drumului de la rădăcina r la un nod x se numește adâncimea nodului x.

Adâncimea rădăcinii este 0.

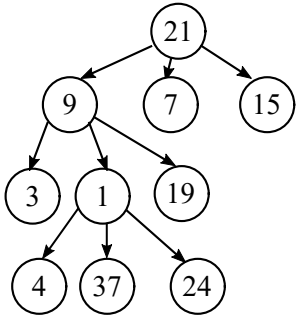
Lungimea celui mai lung drum de la x la o frunză se numește *înălțime* și se notează prin $h(x)$.

Arbore rădăcină



Definiție - subarbore: Se numește *subarbore de rădăcină* x al unui arbore, arborele format din nodul x împreună cu toți descendenții săi.

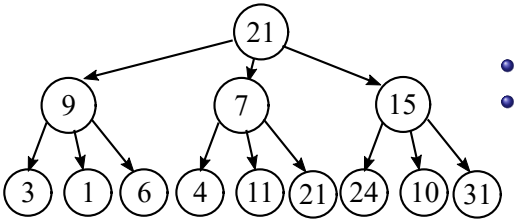
Arbori *n*-ari



Arbori n-ari - definiții. Se numește **arbore** *n*-**ar** un arbore pentru care fiecare nod are cel mult *n* fii.

- *arbore plin* - fiecare nod intern are exact *n* fii.

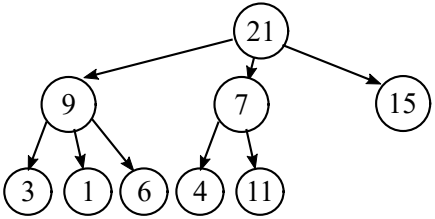
Arbori *n*-ari



Arbori n-ari - definiții. Se numește **arbore** *n*-**ar** un arbore pentru care fiecare nod are cel mult *n* fii.

- *arbore plin* - fiecare nod intern are exact *n* fii.
- *arbore perfect* - fiecare nod intern are exact *n* fii și toate frunzele au aceeași adâncime.

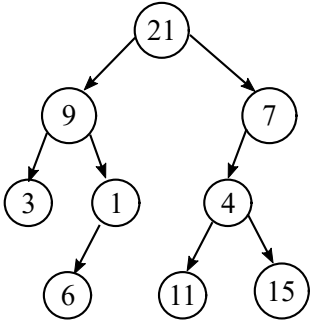
Arbori *n*-ari



Arbori n-ari - definiții. Se numește **arbore** *n*-**ar** un arbore pentru care fiecare nod are cel mult *n* fii.

- *arbore plin* - fiecare nod intern are exact *n* fii.
- *arbore perfect* - fiecare nod intern are exact *n* fii și toate frunzele au aceeași adâncime.
- *arbore complet* - toate nodurile interne, cu excepția eventual a celor de pe penultimul nivel, au exact *n* fii, iar nodurile de pe ultimul nivel sunt așezate cel mai la stânga posibil pe nivelul respectiv.

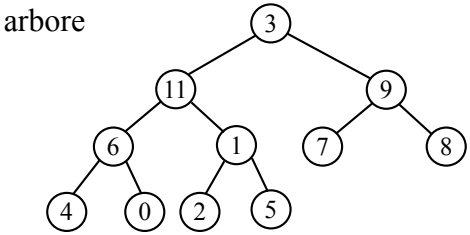
Arbori binari



Definiție - Arbore binar: Un arbore binar este un arbore în care fiecare nod are cel mult doi fii.

Atunci când fiecare nod are 0 sau 2 fii, arborele se numește *arbore binar strict*.

Reprezentarea arborilor binari

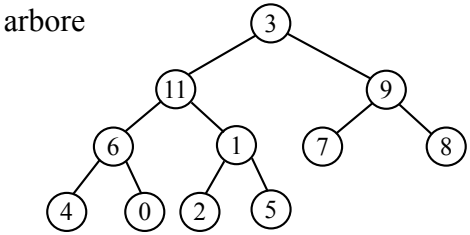


1. Reprezentarea secvențială:
printr-un vector - **array**.

Rădăcina - pe poziția 0.
Pentru nodul de pe poziția *i*:

- fiul stâng - pe poziția ?
- fiul drept - pe poziția ?
- părintele - pe poziția ?

Reprezentarea arborilor binari



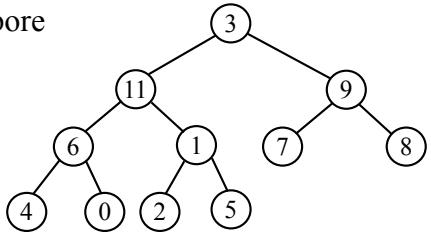
1. Reprezentarea secvențială:
printr-un vector - **array**.

Rădăcina - pe poziția 0.
Pentru nodul de pe poziția i:

- fiul stâng - pe poziția $2 * i + 1$,
- fiul drept - pe poziția ?
- părintele - pe poziția ?

Reprezentarea arborilor binari

arbore



reprezentare prin vector

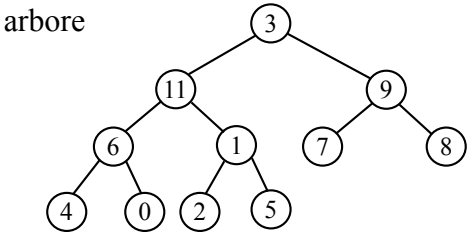
3	11	9	6	1	7	8	4	0	2	5

1. Reprezentarea secvențială:
printr-un vector - **array**.

Rădăcina - pe poziția 0.
Pentru nodul de pe poziția i:

- fiul stâng - pe poziția $2 * i + 1$,
- fiul drept - pe poziția $2 * i + 2$.
- părintele - pe poziția ?

Reprezentarea arborilor binari

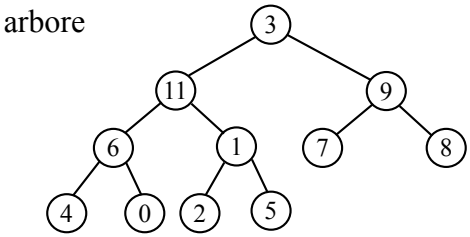


1. Reprezentarea secvențială:
printr-un vector - **array**.

Rădăcina - pe poziția 0.
Pentru nodul de pe poziția i:

- fiul stâng - pe poziția $2 * i + 1$,
- fiul drept - pe poziția $2 * i + 2$.
- părintele - pe poziția $(i - 1)/2$

Reprezentarea arborilor binari



1. Reprezentarea secvențială:
printr-un vector - **array**.

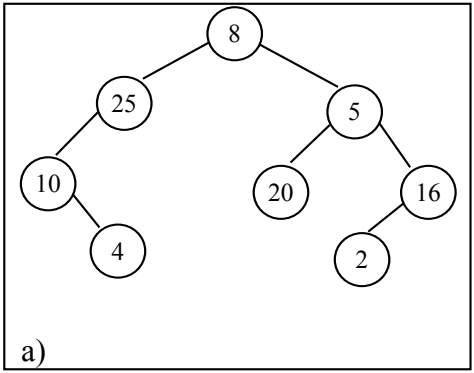
Rădăcina - pe poziția 0.
Pentru nodul de pe poziția i:

- fiul stâng - pe poziția $2 * i + 1$,
- fiul drept - pe poziția $2 * i + 2$.
- părintele - pe poziția $(i - 1)/2$

Utilizare: pentru arbori compleți (de exemplu în cazul heap-urilor).

Reprezentarea arborilor binari

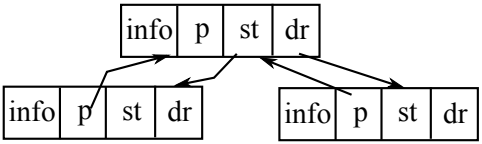
2. Reprezentare mixtă



b)

indice	info	left	right
0	8	1	2
1	25	3	*
2	5	4	5
3	10	*	6
4	20	*	*
5	16	7	*
6	4	*	*
7	2	*	*

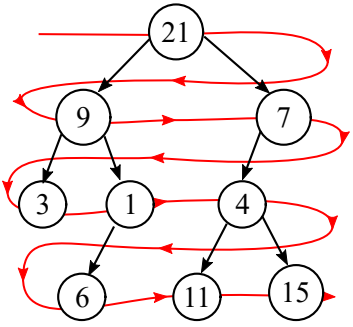
Reprezentarea arborilor binari



3. Reprezentare printr-o structură de pointeri:

```
struct nod{
        int info;
        nod* stanga;
        nod*dreapta;
        nod* parinte;
};
struct arbore_binar{
        nod* varf;
        //metode asociate
};
```

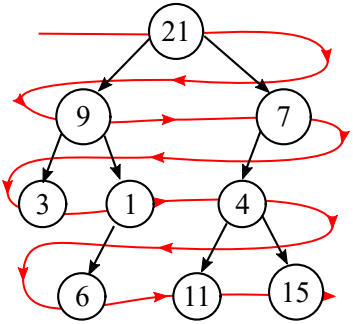
Parcurgerea arborilor binari



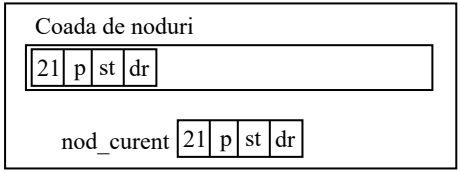
I. Parcurgerea în lățime:

- presupune parcurgerea pe rând a fiecărui nivel de la stânga spre dreapta.
- se poate realiza practic utilizând o coadă C

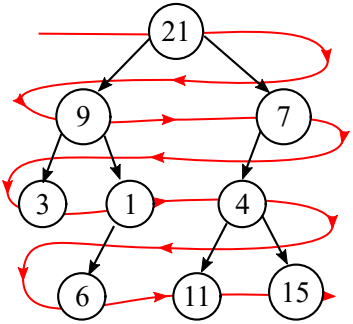
Parcurgerea arborilor binari



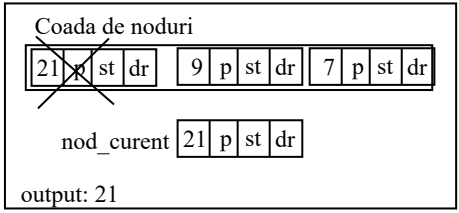
I. Parcurgerea în lățime - Exemplu:



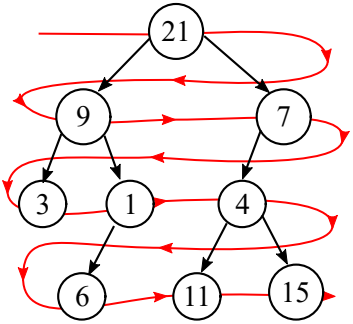
Parcurgerea arborilor binari



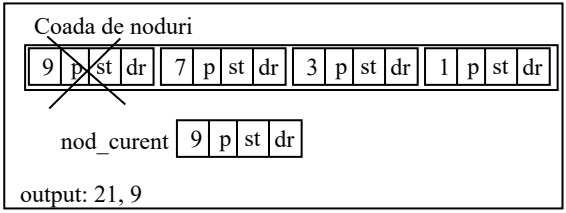
I. Parcurgerea în lățime - Exemplu:



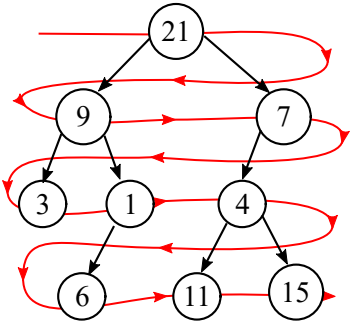
Parcurgerea arborilor binari



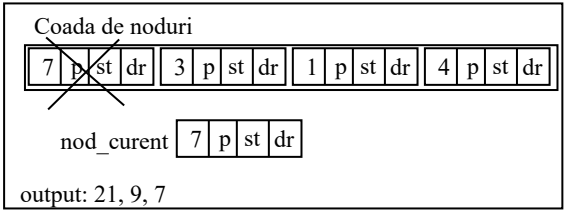
I. Parcurgerea în lățime - Exemplu:



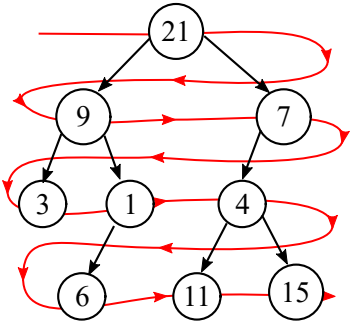
Parcurgerea arborilor binari



I. Parcurgerea în lățime - Exemplu:



Parcurgerea arborilor binari



I. Parcurgerea în lățime - Exemplu:

Coada de noduri

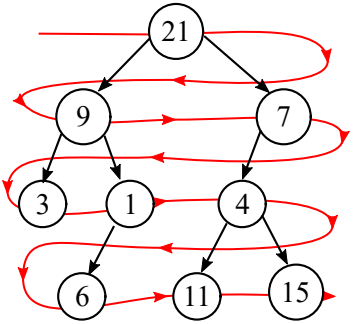
3	p	st	dr	1	p	st	dr	4	p	st	dr

nod_curent

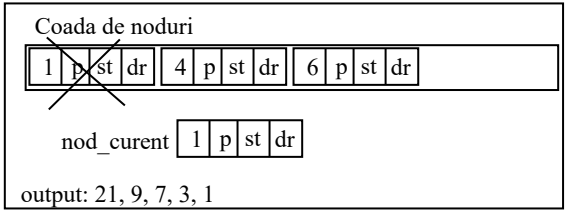
3	p	st	dr

output: 21, 9, 7, 3

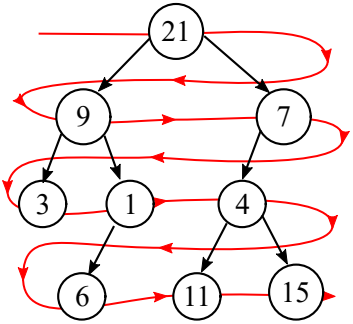
Parcurgerea arborilor binari



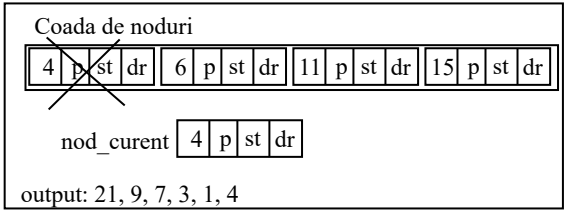
I. Parcurgerea în lățime - Exemplu:



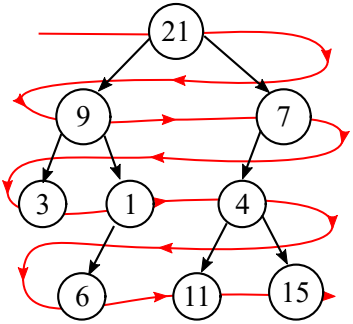
Parcurgerea arborilor binari



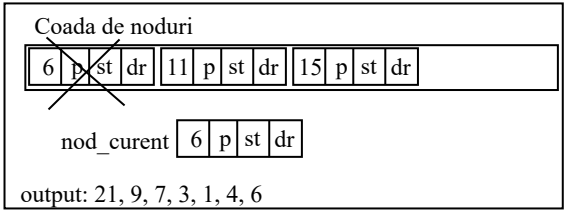
I. Parcurgerea în lățime - Exemplu:



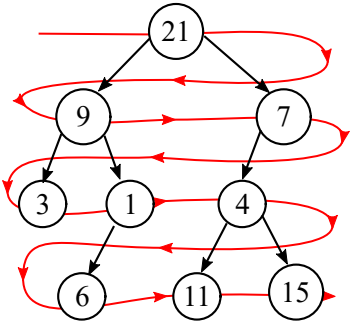
Parcurgerea arborilor binari



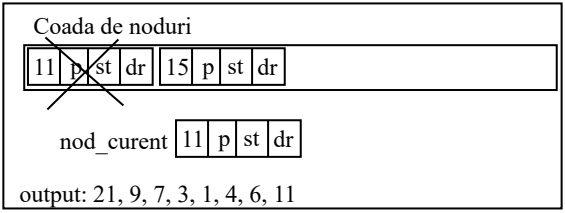
I. Parcurgerea în lățime - Exemplu:



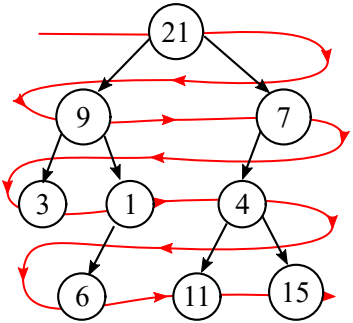
Parcurgerea arborilor binari



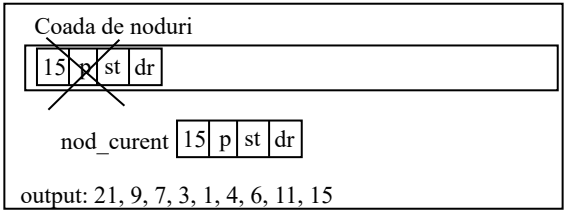
I. Parcurgerea în lățime - Exemplu:



Parcurgerea arborilor binari



I. Parcurgerea în lățime - Exemplu:



S-a golit coada. S-a terminat parcuregerea.

Percuregerea în lățime - pe niveluri

Algoritm 1: Latime

Intrare: Arborele T

Iesire: parcurgerea în lățime

Declaram coada C

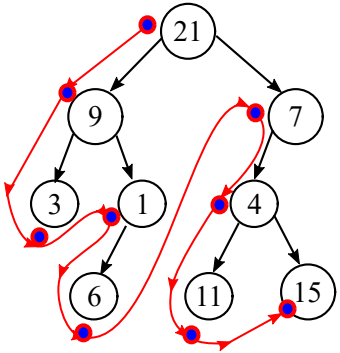
$C.PUSH(T.rad)$

cat_timp $!C.empty()$ **executa**

- $nod\_curent \leftarrow C.TOP()$
- $C.POP()$
- scrie $nod\_curent.info$
- **daca** $nod\_curent.st \neq NULL$ **atunci**
  - $C.PUSH(nod\_curent.st)$
- **sfarsit_daca**
- **daca** $nod\_curent.dr \neq NULL$ **atunci**
  - $C.PUSH(nod\_curent.dr)$
- **sfarsit_daca**

sfarsit_cat_timp

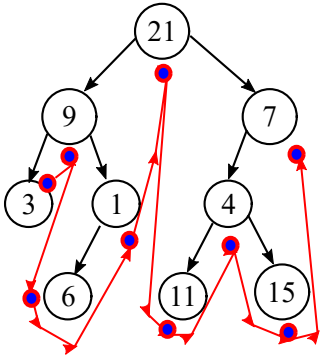
Parcurgerea arborilor binari



II. Parcurgerea în adâncime În funcție de ordinea în care parcurg rădăcina și subarborii, se disting trei tipuri de parcurgere:

- **Preordine (RSD)**: rădăcină, subarbore stâng, subarbore drept: 21, 9, 3, 1, 6, 7, 4, 11, 15

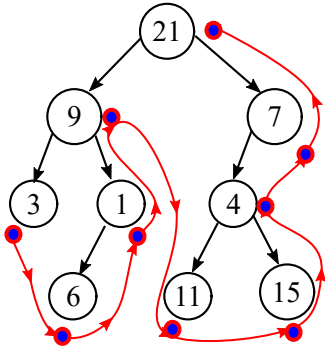
Parcurgerea arborilor binari



II. Parcurgerea în adâncime În funcție de ordinea în care parcurg rădăcina și subarborii, se disting trei tipuri de parcurgere:

- **Preordine (RSD)**: rădăcină, subarbore stâng, subarbore drept: 21, 9, 3, 1, 6, 7, 4, 11, 15
- **Inordine (SRD)**: subarbore stâng, rădăcină, subarbore drept: 3, 9, 6, 1, 21, 11, 4, 15, 7

Parcurgerea arborilor binari



II. Parcurgerea în adâncime În funcție de ordinea în care parcurg rădăcina și subarborii, se disting trei tipuri de parcurgere:

- **Preordine (RSD)**: rădăcină, subarbore stâng, subarbore drept: 21, 9, 3, 1, 6, 7, 4, 11, 15
- **Inordine (SRD)**: subarbore stâng, rădăcină, subarbore drept: 3, 9, 6, 1, 21, 11, 4, 15, 7
- **Postordine (SDR)**: subarbore stâng, subarbore drept, rădăcină: 3, 6, 1, 9, 11, 15, 4, 7, 21

Arbori - Utilizare

- Sortare: Heap-sort

Arbori - Utilizare

- Sortare: Heap-sort
- Compilatoare: arbori sintactici de derivare

Arbori - Utilizare

- Sortare: Heap-sort
- Compilatoare: arbori sintactici de derivare
- Procesare de imagine / grafică: arbori Quad, arbori PR

Arbori - Utilizare

- Sortare: Heap-sort
- Compilatoare: arbori sintactici de derivare
- Procesare de imagine / grafică: arbori Quad, arbori PR
- Clasificare: arbori de decizie

Arbori - Utilizare

- Sortare: Heap-sort
- Compilatoare: arbori sintactici de derivare
- Procesare de imagine / grafică: arbori Quad, arbori PR
- Clasificare: arbori de decizie
- Dicționare, căutare eficientă: arbori de căutare.